STEMSEL Beginners Project 9: Cool Fan Hat

Problem
In this project we want to use our STEMSEL kits to design a hat that can keep us cool in the sun, and also looks cool.

Background
When we head out into the sunshine, it is important to think about protecting ourselves. The sun emits UV rays which can cause sunburn and even skin cancer. For this reason we should always slip-slop-slap when playing or working outside. But can we do better than this?

Ideas
What can we use from our kits to keep us cool? How can we liven up our hats to make them look cooler? Do we want the hat to be on all the time, or only when we are in the sun? If we have more than one sequence of light flashes, can we change the number or times each sequence plays? Have we done something like this before in an earlier project.

Plan
The fan can be used to keep us cool, but we will only need it to be on when we are in the sun. Therefore we can use an LDR that will sense the bright sunshine then turn on the fan. We can also use the LEDs flashing in a sequence to make our hat look cooler.

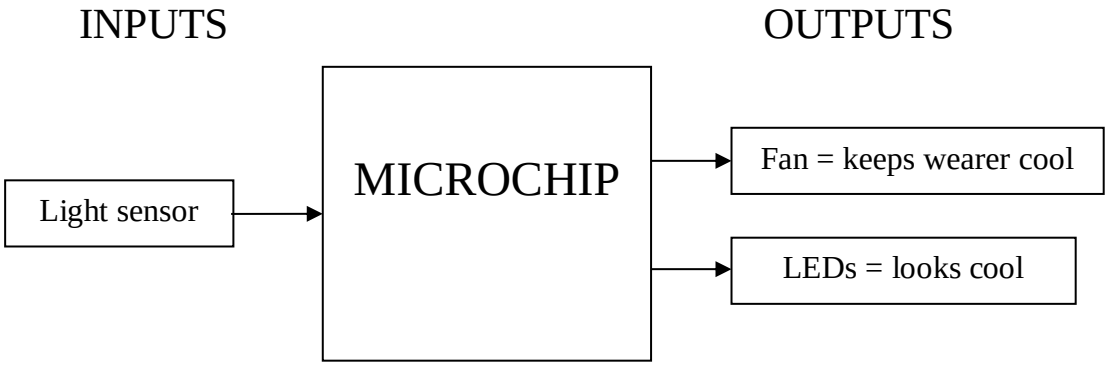

Figure 1

The hat should check to see if we are in the sun, and if so it should turn on the fan and LEDs until we go inside again. Just like in previous projects, we can check the brightness using a threshold value. If we want to have more than one sequence we can have a counting loop like we used in the washing machine project to set how many times each LED sequence plays.

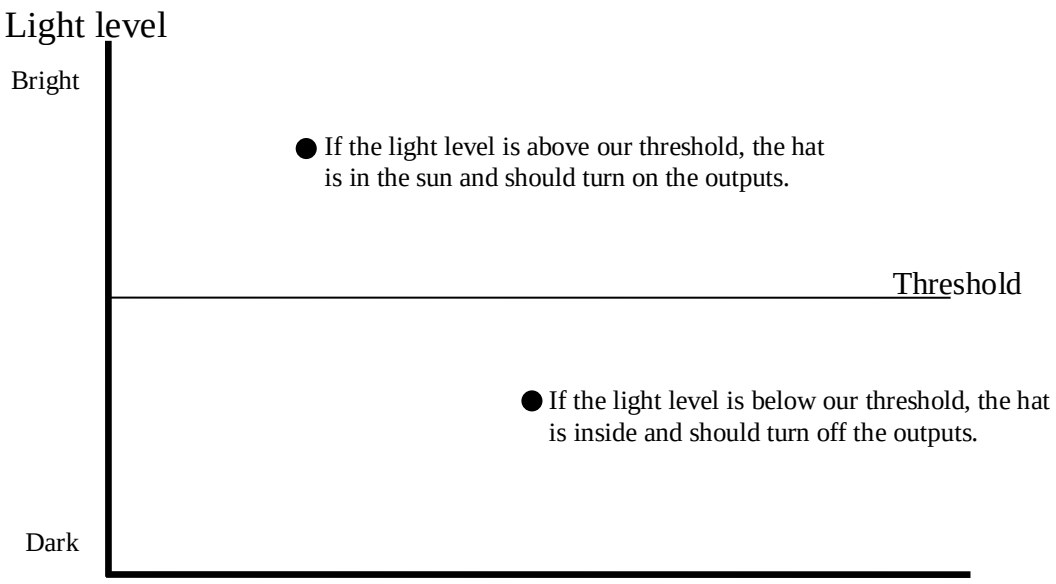

Figure 2

Design the Circuit

Open ezCircuit Designer and add the components we chose in the plan.

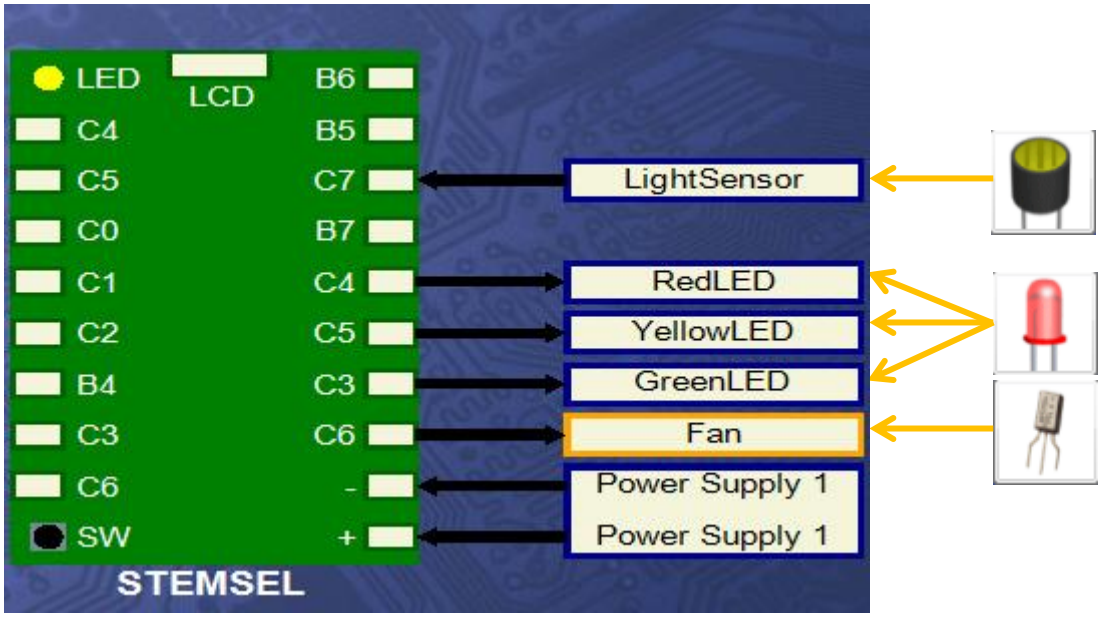

Figure 3

Build the Circuit

Use the ezCircuit Designer I/O diagram to connect the hardware. Remember that black wires go into negative, red wires go into positive, and white wires go into the pin specified in the design. For the fan, one white wire should go into C6, and the other into negative.

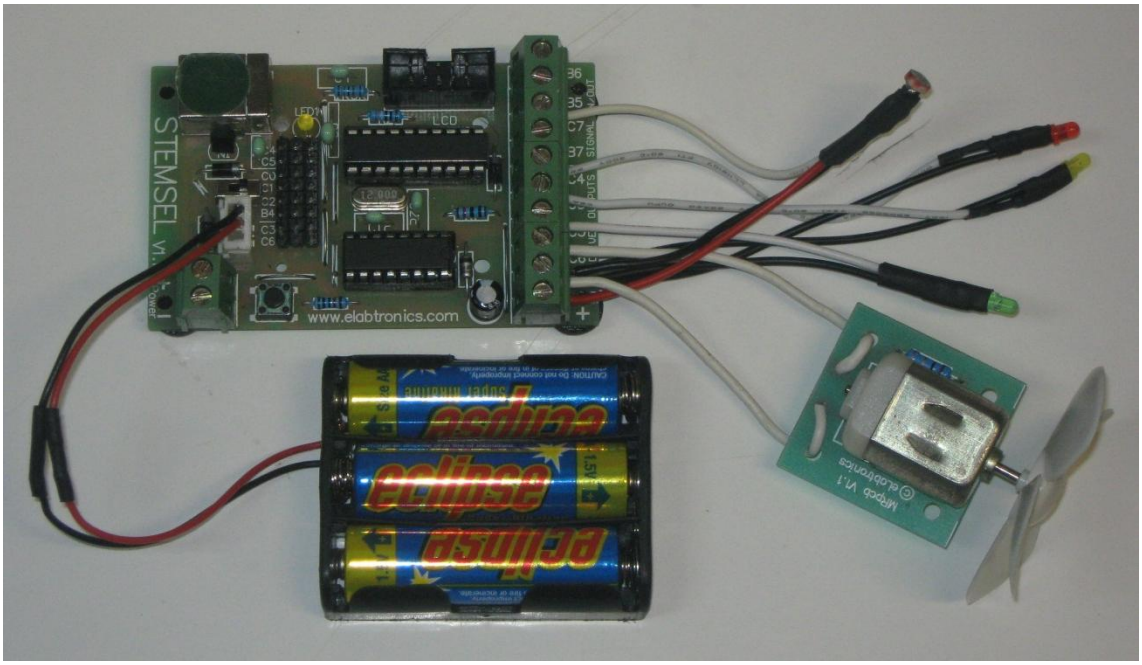

Figure 4

Programming

Use CoreChart to program the functions of the hat. After you have used the test routines to check the outputs, delete them so that we can start writing our own program.

1. According to the plan, the first thing to do is to check if the hat is in the sun or not. Use an AnalogIn icon to read the value from the LDR.

2. Use a Compare icon to compare the value from the LDR with a threshold value, say 3V. Tick both the Above and Below checkboxes.

3. The graph in the plan tells us that if the light level is below the threshold, we are inside and want turn off the fan and LEDs. Use OnOffPin icons to turn off all our outputs, then group them so they all fit beneath the Below Decision icon.

4. If the light level is above 3V, then we are outside and want to turn on the fan and LEDs. Use an OnOffPin icon to turn the fan on first, then use sequences of OnOffPin icons and TimeDelay icons to make your own LED sequence. You should remember this from the Traffic Light Project.

5. Group these icons together so that they all fit beneath the Above Decision icon. Add a GoTo START at the end of the program, then send the program to chip to test it. Did your LED sequence work how you wanted it to?

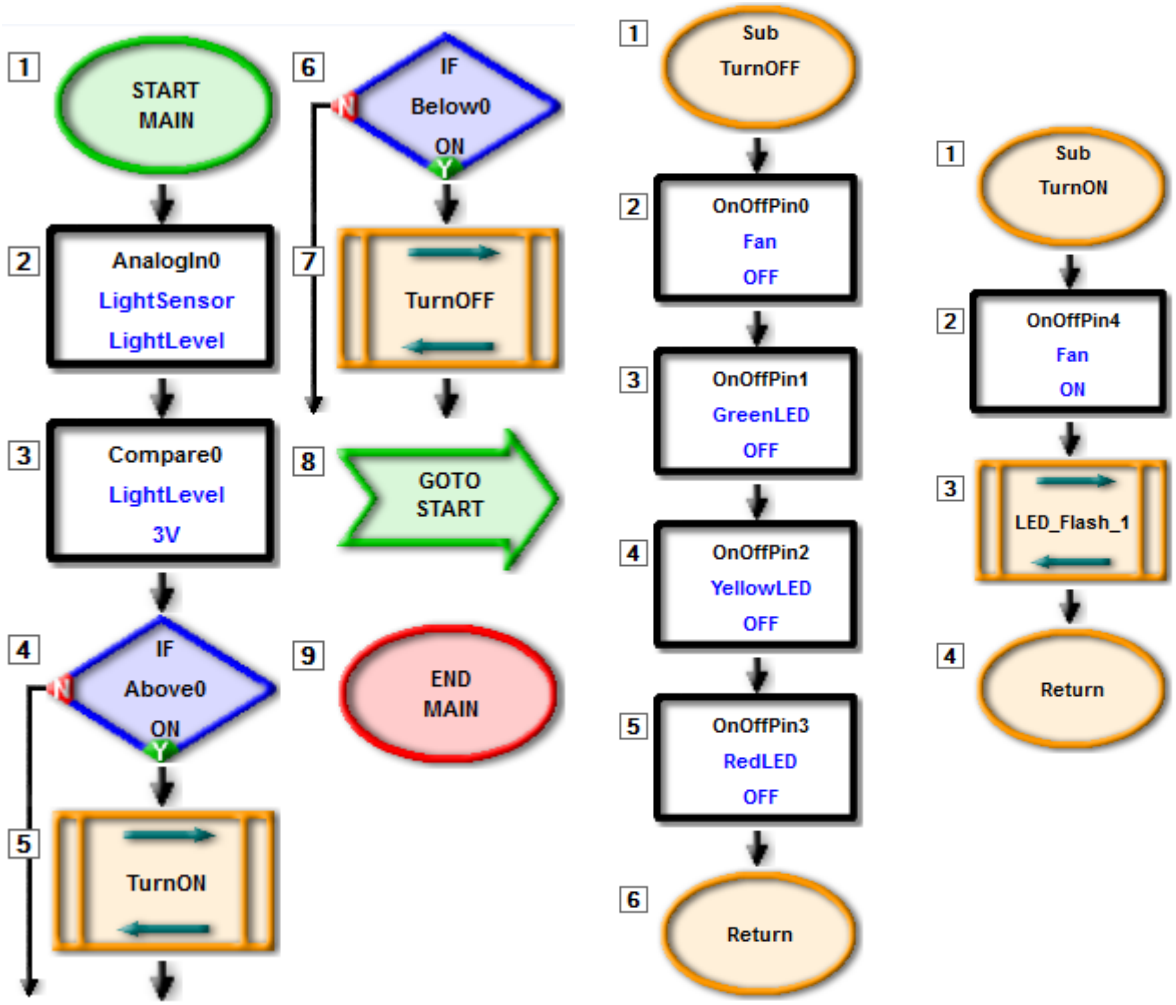

Figure 5

Extension

Can we make our hat look even better by having several different sequences? Is there an easy way to make the program do these sequences more than once? In our plan we decided we could use loops to set how many times each sequence plays, like with the wash cycle in the washing machine project. In this project we will change it slightly to a decrementing loop.

1. If you haven't already, group your LED sequence into one group.

2. Think up and program a second sequence, then group it separately.

3. If we wanted to make these sequences flash a certain number of times each, we could just put in that many groups, but what if we wanted to change it later, or used that sequence lots of times. Is there an easier way? First, click Numbers, then select Set_Number from the Icon Properties list. Place the new icon just before your first sequence.

4. Use the Set_Number icon to set a variable called Count to the number of times you want your first sequence to run.

5. Put an address below the Set_Number icon, then a Subtract icon below your sequence. Use the Subtract icon to take away 1 from Count, saving the result as Count again.

6. Put a Compare icon below the Subtract, comparing Count with 0. Tick the Above box.

7. Modify the automatically generated GoTo icon to loop back to the address you added before the first sequence.

8. Do the same for the second sequence. You can use the same variable Count, but use a different address.

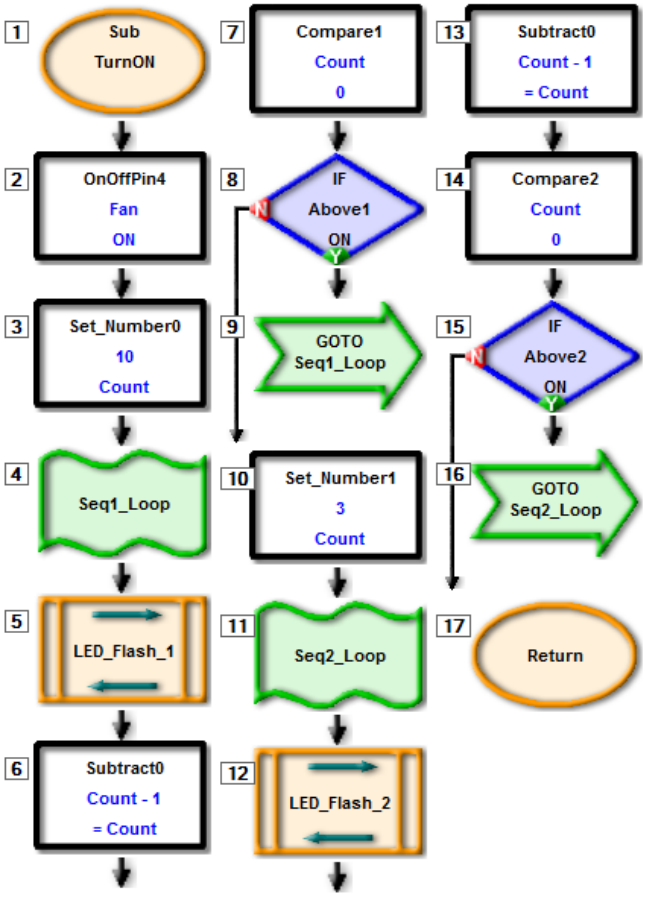

Figure 6

These can be grouped together to make the program look neater. Have a play around and see what kind of sequences you can create.

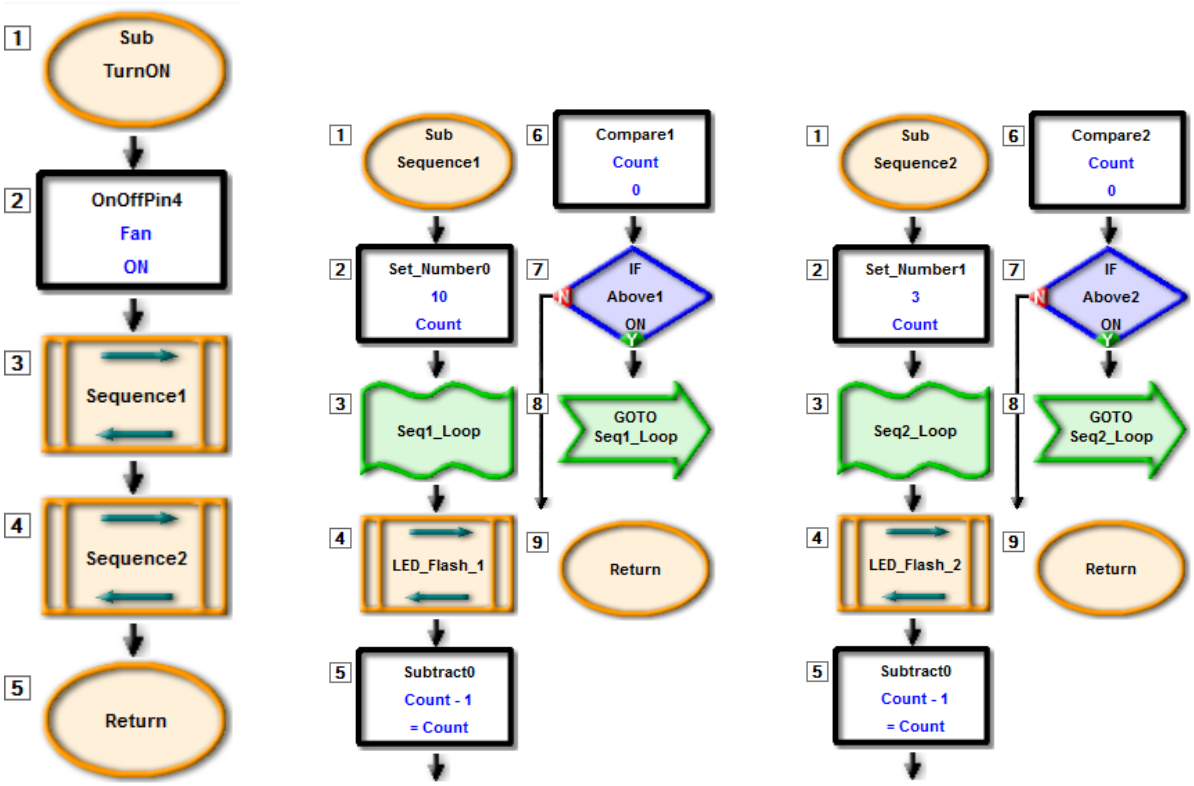

Figure 7

Summary

In this project, we made a fan hat that can keep us cool, but also looks cool because of the LEDs we programmed to flash. You should now understand more about sequencing, including how to use decrementing loops to perform certain tasks a set number of times. Although it is a bit more work to set up, it is easier and neater than pasting lots of groups into the program, and also makes it much easier if you want to change the number of times each sequence flashes, since you only have to change the Count value.

