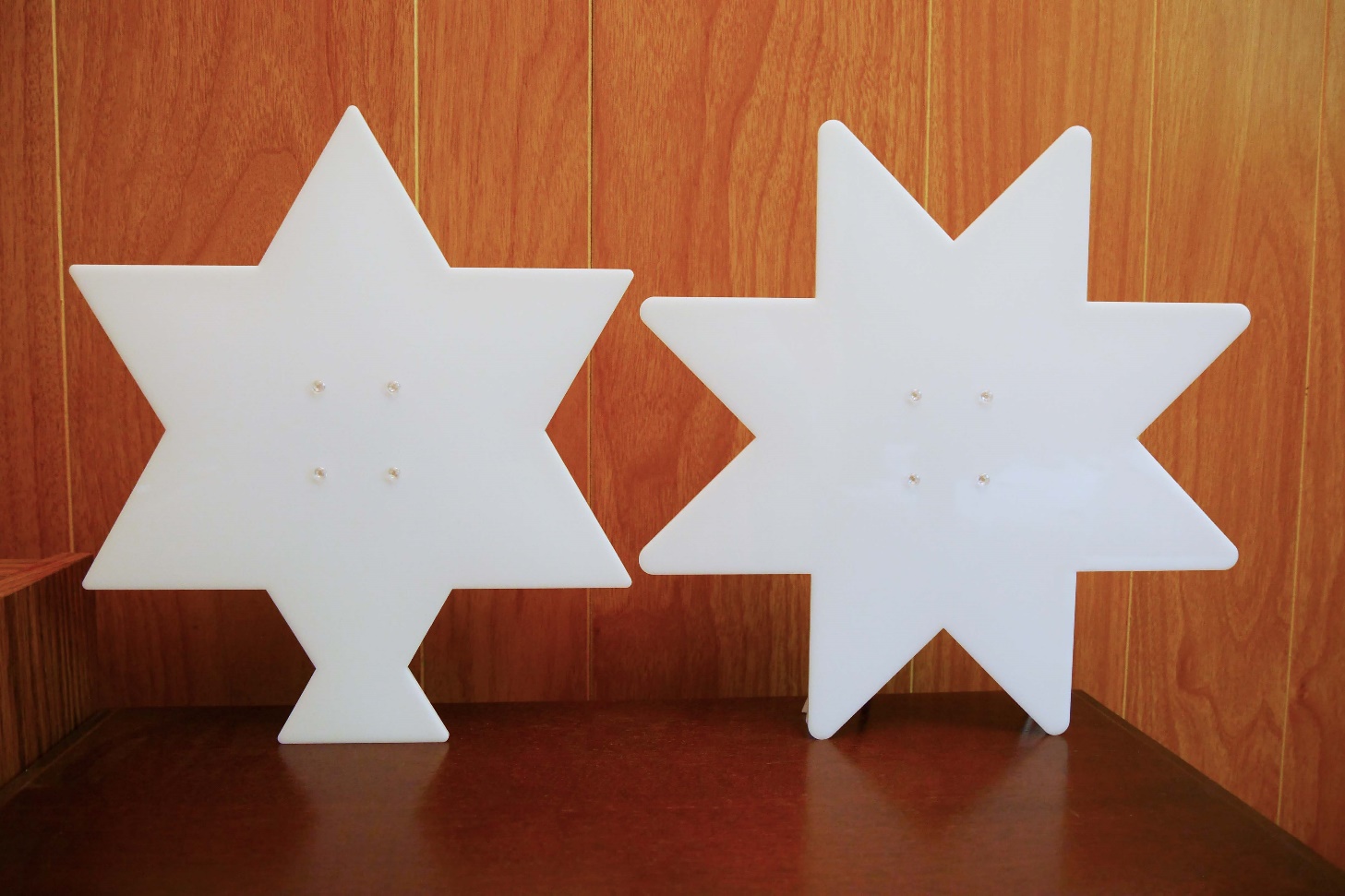
Grant Grummer

8/25/2020

Starlite User’s Manual



Revision: A.0

Table of Contents

[Introduction 1](#_Toc50456092)

[Starlite Design Environment 1](#_Toc50456093)

[Quick Custom Displays 2](#_Toc50456094)

[Programming Starlite 2](#_Toc50456095)

[Exploring and/or Editing the Starlite design 2](#_Toc50456096)

[Run Synthesis 3](#_Toc50456097)

[Run the Build 3](#_Toc50456098)

[Block Diagram of Starlite Code 4](#_Toc50456099)

# Introduction

Welcome to the Starlite User’s Manual. Starlite can be constructed as a 6 or 8-point star. They’re approximately 10 inches tall. They consist of outer and inner translucent acrylic star shapes, tri-color LED strips, a circuit board, and the hardware to connect them together. An optional 3D printed back cover can also be made, for a more finished look.

Starlite uses the tri-color LED strips to produce different light displays. Starlite is programmable, so the displays are only limited by your imagination. A set of default light displays can be viewed here:

<https://drive.google.com/file/d/1XsCNBrU7cHPyYp-3UZcMKzgj40U59saX/view?usp=sharing>

A small set of display specific instructions are used to create different display patterns. There are up to 16 possible instructions. Not all of these instructions are defined at the time of this writing and are available for those who wish to dive deeper into this project.

The circuit board contains all the electronic hardware needed to drive the LED strips. The brain of the hardware is a Lattice iCE40 Ultra Plus 5K FPGA (Field Programmable Gate Array). It, along with programming and other support logic, is on the [UPduino v3.0](https://www.tindie.com/products/tinyvision_ai/upduino-v30-low-cost-lattice-ice40-fpga-board/?pt=ac_prod_search) low cost daughter board. It’s mounted onto the StarLiteMain board. The StarLiteMain board contains switches, power connector and interface hardware.

Starlite is controlled via three switches. The power switch enables/disables power from the power input micro USB connector. The upper pushbutton switch selects one of four display instruction sets. Each set can have up to 256 instructions. The lower pushbutton switch selects different brightness levels.

# Starlite Design Environment

In order to program Starlite, one must first setup a design environment. Refer to the “Starlite Installation and Setup” document for instruction on how to set it up. The rest of this document assumes this step has been completed.

# Quick Custom Displays

A quick way to get started with making your own custom displays is to use the default instruction set to build new display instruction sets. One doesn’t need to know anything about FPGAs to do this. Use the supplied “LED\_StarLiteUP\_Instruction” spreadsheet to create each instruction. Use a text editor to combine the new instructions into a set of instructions. The text editor needs to be able to select columns of text like “[Notepad++](https://notepad-plus-plus.org/downloads/)”. Use the supplied example display instruction sets, in the “patterns” folder, to see how this works. Remember to end each instruction set with the Finish instruction!

Once an instruction set has been completed, it needs to be split in half vertically. Refer to the supplied example display \*\_hi.mem and \*\_lo.mem instruction sets to see how the split looks. Splitting the instructions is necessary because the FPGA memories are 16-bits wide, while instructions are 32-bits wide.

Replace the current instruction set(s) with your new instruction set(s) in the “instr\_ram\_full6.list” or “instr\_ram\_full8.list” (depending on whether you’re building the 6 or 8-point star). Then run “memloader6.bat” or “memloader8.bat” batch file. Once that completes successfully, run “bitmapper.bat”.

# Programming Starlite

There are many programmers that can be used to program Starlite. One is the Radiant [Programmer Standalone](https://www.latticesemi.com/Products/DesignSoftwareAndIP/FPGAandLDS/Radiant) and the other is the Diamond [Programmer Standalone](http://www.latticesemi.com/en/Products/DesignSoftwareAndIP/ProgrammingAndConfigurationSw/Programmer.aspx#_20C94305815A4B3AAAFEA8B83943B751)). They’re both very similar. The following steps can be used with the Radiant Programmer to program Starlite. The Diamond Programmer steps may be slightly different.

1. Connect the micro USB port on the UPDuino v3.0 to your PC

2. Open the Radiant Programmer

3. Click `Detect Cable` then `OK`

4. After scanning, select `Generic JTAG Device` and `Select iCE40 UltraPlus`

5. Under `Device` click iCE40UP3K and change it to iCE40UP5K

6. Under `Operation` double click `Fast Program` and change `Target Memory: ` to `External SPI Flash Memory`

7. Select your `\*.hex` programming file under `Programming file`.

8. Configure the following `SPI Flash Options`

a. Winbond

b. W25Q32JV

c. 208mil 8-pin SOIC

9. Click `Load from File` under `SPI Programming` to get load size

10. Click OK

11. Click `Run` -> `Program Device`

# Exploring and/or Editing the Starlite design

The code base for Starlite is contained in the download (clone) from GitHub (as explained in “LED\_StarLiteUP\_Instruction”). Feel free to explore the design and customize it to your liking. If changes were made and you’d like to build the design, synthesis will need to be run first.

## Run Synthesis

Double click “Run Lattice …”. Once it completes successfully, move on to the next section.

## Run the Build

Right click on “Run P&R”. Select “Run Router”. This will run the steps before “Run Router” and “Run Router” too. There’s no need to run “Bitmap” at this point. Once these complete successfully, it’s time to insert display patterns. This is covered in the previous Quick Custom Displays section. Refer to it for directions. “Bitmap” can be run using the “bitmapper.bat” file outside of iCEcube2 (as noted in the Quick Custom Displays section) or one can run “Bitmap” in iCEcube2. This completes the build. Refer to the previous Programming Starlite section to program Starlite.

# Block Diagram of Starlite Code

