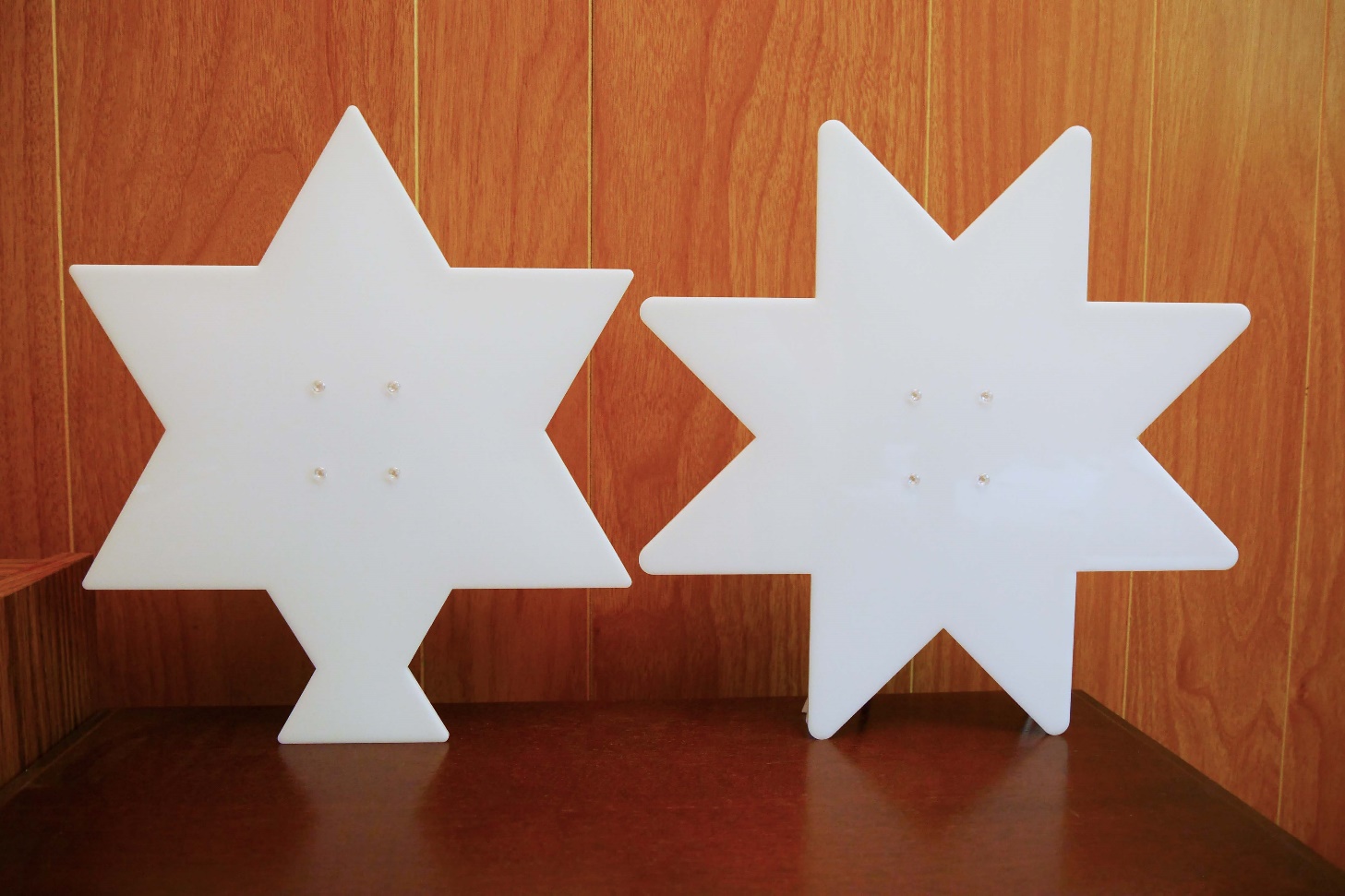
Grant Grummer

8/25/2020

Starlite User’s Manual



Revision: A.0

Table of Contents

[Introduction 1](#_Toc49261664)

[Starlite Design Environment 1](#_Toc49261665)

[Quick Custom Displays 1](#_Toc49261666)

[Programming Starlite 2](#_Toc49261667)

[Exploring and/or Editing the Starlite design 2](#_Toc49261668)

[Run Synthesis 2](#_Toc49261669)

[Start the Build 2](#_Toc49261670)

[Inserting the Patterns 2](#_Toc49261671)

[Block Diagram of Starlite Code 3](#_Toc49261672)

# Introduction

Welcome to the Starlite User’s Manual. Starlite can be constructed as a 6- or 8-point star. Both are slightly over 10 inches tall. They consist of outer and inner translucent acrylic stars 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 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 commands are used to create different display patterns. There are up to 16 possible commands. Not all of these commands 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. That is mounted onto the StarliteMain board. It 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 command sets. Each set can have up to 256 commands. 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 command sets. Use the supplied “LED\_StarLiteUP\_Instruction” spreadsheet to create each command. Use a text editor to combine the new commands into a set of commands. 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 command sets, in the “patterns” folder, to see how this works. Remember to end each command set with the Finish command!

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

Replace the current command set(s) with your new command 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. Plug the UPDuino v3.0 in to your PC using the micro USB port on the board

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.

## Start 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”, will do that in the next section. Once these complete successfully, move on to the next section.

## Inserting the Patterns

Instructions on how to insert display patterns into the design are covered in this section. This is done outside of iCEcube2. Actually, there’s no need to return to iCEcube2 to complete the build, but one can if desired. Refer to Quick Custom Displays section for instructions.

# Block Diagram of Starlite Code

