# Approvals

|  |  |
| --- | --- |
| Software Design Engineer Date | Manufacturing Engineer Date |
|  |  |
| Quality Representative Date | Other Date |
|  |  |

# Summary of Changes

|  |  |  |
| --- | --- | --- |
| **Revision Level** | **Prepared By** | **Description of Change** |
| A | G. Chopcinski | Initial Release. |
|  |  |  |

# Pre-Build Instruction overview

The ASL165 firmware is built on a Windows-based PC. You may need a powerful PC running Windows 10 or later.

# Build Environment

The Windows-based PC used to develop the ASL165 firmware is running Windows 10. I suggest getting a machine with at least 4 Giga Bytes of RAM and substantial CPU horsepower. The development tool, Renesas E2Studio, is a memory hog and power hungry.

The Renesas Web-site has good information to setup the e2studio for development. Refer to the following link: <https://synergygallery.renesas.com/media/products/2/374/en-US/r20ut4204eu0120-synergy-e2-studio-getting-started-guide.pdf> for further details.

# Source File information

The source code is maintained in a GitHub site at <https://github.com/garychop/ASL_Display.git>. This site is cloud-based and requires an account and an “invite to participate” to access the source code.

# Build Support information

All files necessary to build the ASL165 Display/HHP firmware executable is included in the GitHub repository mention above.

The application is built using E2Studio. This is an IDE built on the Eclipse framework. The version to build this application is e2studio version 7.5.1 with the Synergy System Package (SSP) version 1.7.0.

As a note, if installing your PC from scratch, you may have to use the e2studion File -> Import feature to properly setup the e2Studio navigation.

## No Release build?! What!?

The e2studio build has capability to build a “debug” and “release” version of the firmware. According to Renesas documents, the difference lies in the “.elf” files which are used for debugging purposes. The “S record” formatted files are the same regardless of build configuration. They suggest to simply use the “Debug” build configuration and use the resulting S-Record to program the micro’s in production. Refer to the following link for details.

<https://renesasrulz.com/e2studio/f/e2studio-forum/6919/compiling-a-release-with-estudio>

# Build instructions

Once the e2studio is properly setup, build the firmware using **Project -> Clean…** menu item. If successful, there will be no errors but there will be 14 warnings. These warning are generated while compiling the supporting files and NOT our source files. Therefore, we cannot resolve all warnings.

# Deployment

The production micro’s can be programmed using a Segger-source J-Link Debugger with an 20-10 pin adapter and the J-Link Flash program from Segger.

 

<https://www.segger.com/products/debug-probes/j-link/> is the link for the J-Link Flash code.

Follow the instructions below to program a production ASL165.

* Locate the ASL\_Display.srec and put into the same directory as the J-Link Flash program. This file is in the ASL Display GitHub repository.
* Retrieve the ASL110\_Display.jflash file from the GitHub repository in the “Production Programming” folder and put in the same directory as the J-Link Flash program.
* Start the J-Link Flash program (JFlash.exe).
* Open an existing project named ASL110\_Display.jflash. This should set up the MCU and the file ASL110\_display.srec to program.
* Then program using F7. (Menu "Target" -> "Production Programming F7".

This should erase, program and verify the FLASH on the device. Here's what the J-Flash screen looks like when started with the project.

