**Embedded Systems Design Project**

Luis Ferreira, Warren Levesque, Sabrina Melo Goncalo, and Abdul Mueed

ENGN4060

Professor Wai Yung

**Introduction**

Throughout many generations, video games have become a popular hobby. Among that, there is a type of gaming called retro gaming. It is the collection and playing of older personal computers, consoles, and video games- usually ones that have been discontinued or obsolete. Some of these games and consoles are well known. Examples include the Tetris, Street Fight, and Space Invaders games, in addition to the Sega Mega Drive, Nintendo 64, and SNES consoles. During the past school year, a project in relation to retro gaming has been assigned and thus developed. The goal of this project was to be able to collectively create a portable gaming device using a DE10-LITE board, as well as an Arduino board that would be able to allow the player to select, deselect, and create matches of icons while keeping track of score and rounds. However, since retro gaming has been in the market for a very long time, it would be illegal and considered piracy to simply copy it. In order to avoid doing so and potentially harming the original creator of the game by causing a loss in sales, skills gained throughout engineering courses had to be applied by team members. These skills include the design of an enclosure for a FPGA board, ability to program in C & Arduino, & designing icons to be used in BMP format.With four main focused sections including design, automation, DE10-LITE programming, and Arduino programming, the game was built entirely from scratch. This was not an easy project to say the least but overall was very enjoyable because we were all proud of the outcome. Many hurdles and challenges were encountered throughout the project, but solutions were found.

**Summary of Work**

Since it was a group of four, the project was divided into sections. The sections included the design, automation, Arduino programming and DE10-LITE programming . It was essential to use both the Arduino board and the DE-10 board because the DE-10 was primarily used to handle the graphics while the Arduino handles most of the logic portion. These two boards were to communicate back and forth to display icons on a VGA display screen. Nios II for Eclipse was the integrated development environment (IDE) used for this project. Thirty-six icons were displayed in a 6x6 grid. The supported resolution of the DE-10 board is 640 horizontal pixels by 480 vertical pixels, but only half of it was used for the icons. Making sure the positioning was accurate was vital. Navigating the screen with the cursor was a key feature of this game. Furthermore, a joystick was implemented to move the cursor on the screen, and a push button was used to select, deselect, and remove an icons after being matched. There are many details to look at in this project but one the most important was the built circuit. When incorporating user input through the joystick, a way to read the data was necessary. For this, the onboard Arduino analog to digital convertor was used . This allowed it to receive the analog data through the joystick and convert it to digital data to be used in the program in order to tell the orientation of the joystick. When it came to the enclosure design, all group members were required to have a design, but only one would be used.

**Project Plan**

The plan for this project was based on a weekly check-in system. Since it was a two semester long project, both semesters included different completion parts of both the design and programming. Table 1 shows the first semester weekly progress, while Table 2 shows the second semester weekly progress.

Table 1

| **Fall Semester Project Plan**  **( 9/30/21-12/19/21)** | **Weekly Activities** | **Weekly Achievements** |
| --- | --- | --- |
| **Week 1** | -Class orientation |  |
| **Week 2** | -Start of project |  |
| **Week 3** | **-**team made final design pick | **-**decided on spiderman theme |
| **Week 4** | **-**developed 32x32 icons  -developed letters and number  -worked on automation code | **-**completed automation code to convert BMP’s into arrays stored in the header. |
| **Week 5** | **-**worked on joypad movement code  -worked on DE10 code | **-**took icons from header and displayed them all on screen. |
| **Week 6** | **-**selected best 7 icons  -started communication code between Arduino and DE10 | -generated 6x6 list and assigned numbers to each icon. |
| **Week 7** | -continued communication code between Arduino and DE10 | **-**created teams github  -selected which enclosure design to be final. |
| **Week 8** | -continued communication code between Arduino and DE10  -worked on OC flash file | **-**modified the case to fit all boards |
| **Week 9** | -continued communication code between Arduino and DE10 | **-**finialized OC flash file |
| **Week 10** | **-**added pushbutton circuit to arduino for functionality |  |
| **Week 11** | **-**modified OC flash file  -worked on selection function | **-**finalized a function for selecting matched icons |
| **Week 12** | **-**made designs brighter  -started work on deselection of icons | **-**converted all new icons from BMPs into arrays for the header files |
| **Week 13** | **-**continued to work through bugs on the deselection process | **-**finialized enclosure printed  -finished deselection of icons |
| **Week 14** | **-**project check in presentation |  |

Table 2

| **Spring Semester Project Plan (1/10/22-5/3/22)** | **Weekly Activities** | **Weekly Achievements** |
| --- | --- | --- |
| **Week 1** | -class orientation |  |
| **Week 2** | -worked on the ability to clear selected icons |  |
| **Week 3** | -worked on randomly filling the missing gaps of cleared icons with automation code | -finialized clearing the selected icons |
| **Week 4** | -added a round part to update as each match selection cleared and regenerated. | -finalized round section |
| **Week 5** | -added a score part to the code to keep track of how many icon matches the user made | -finalized score section |
| **Week 6** | -worked on bugs and coding errors | **-**fixed numerous graphical bugs |
| **Week 7** | -worked on bugs and coding errors | -presented working game to instructor |
| **Week 8** | -started working on the final presentation of the project. |  |
| **Week 9** | **-**continued to work on the final presentation while touching up minor parts of the code for a final product. |  |
| **Week 10** | -made changes to powerpoint and how the team would be presenting the information | **-**presented and received feedback from instructor |
| **Week 11** | **-**presented again after feedback was received | **-**much better feedback this time around, only small changes needed past this point |
| **Week 12** | **-**started poster and technical writing report | **-**finalized poster design |
| **Week 13** | **-**continued working on technical writing report | **-**finialized technical writing report |
| **Week 14** | - |  |

**Design, Implementation, Building and Testing**

As having a visual is essential to see how the game works, the design of the project was the first part to be implemented. The theme of Spider-Man was decided among teammates, and thus icons, letters and numbers were created using Adobe Photoshop. This was the chosen application for many reasons; it has a variety of colors to choose from, allows the use of layers, allows for drawing individual pixels, and it allows the exporting of a file as a BMP. All of these were critical in both the design process as well as programming, thus making it the best application to use. Ten 32x32 pixel icons were created as the unselected versions. The selected version of the icons were then made by adding layers on top of the original icon design, and drawing spider webs on the top layer. For the keeping of round and score, all of the letters and numbers were 16x24 pixels. Everything was exported as a BMP file, which was used for the automation portion of the project.

Furthermore, a handheld enclosure was designed using OnShape. Its ease of access made it the best choice for the enclosure design. It was designed with the idea of it being ergonomic in mind. It started off as a square with two circular handles for comfortability to hold, and had a more open cover to hold the boards in place. The base also had appropriate holes for the power and VGA cables, as well as an additional hole to be able to access the reset and selection buttons. However, shortly after having 3D printed the first design, it was agreed that it needed to be made thicker and that the edges would be better if they were softer. The holes also had to be fixed since they were not properly measured the first time, and therefore the cables did not fit properly. It was also decided that the cover should be a bit more closed in order to protect the wiring from coming out of place. After a few trials, the enclosure ended up being thick enough to protect the boards itself, as well as very comfortable to hold for playability.

For the automation portion of this project, some of the tasks which would be repeated and create functions that could be called at multiple points were made to reduce the overall size of the code and to reduce the need for menial tasks to be performed by a human. One of these tasks that needed to be automated was the conversion of the BMP file to a usable format. To make this function, python was used. Python was selected because this function did not need to be done within the main program and because python is a higher level programming language, which makes it easier to program in. The BMP format stores data by taking each pixel and storing its color value as a hexadecimal value. Therefore, the BMP file was used because it stores data pixel by pixel in a predictable format, making it easier to create a program that can convert these files. . In Python, these values were extracted from the BMP image and stored in a text file. At first these values were stored in an array format for usage as a header file, but later they were stored in a list format with 8 bit segments for use in the onboard hard drive.

The main part of the code is comprised of two separate codes that communicate with each other. The code starts off with the DE10-LITE portion of the code drawing the initial layout of the game on screen, which includes the title, score, and round portions of the screen. At the end of the initial layout being generated, the DE10 attempts to establish communication between the boards by sending a single value over to the Arduino and if successful, the Arduino sends back another value to confirm. This is important to make sure the two boards are actually able to communicate with each other and so that the two boards can start off from the same point. After sending these values, the Arduino program randomly generates the first set of values that are used to generate the icons on screen, and stores them in a two dimensional array which relate to the position of the icons. Another array is also created based off of which icons can and cannot be selected as a part of a three icon match, which is used later to limit the user to only be able to select matching icons.

With this the two codes enter into their main loops where the actual game is played. The DE10 portion of the code waits for the Arduino to send over a specific value, initiating the collection of information between boards. The Arduino sends data over to the DE10 in six value sections. The first thirty-two values that are sent to the DE10 are ones which relate to the 6x6 display of icons on the screen, which the DE10 receives and processes this function. While processing the data, the DE10 keeps track of which part of the data it is receiving with a counter in order to make sure that the data being received is related to the proper values. The next values that are sent are the x and y position of where the user is on the screen. This data is used to draw a cursor allowing for the user to have a visual representation of where they are. The values for score and round are then sent over and shown on screen to make sure that they are updated properly. Since all of these values do not add up to a multiple of six, two extra values had to be sent over to complete the final set of values.

The Arduino also handles the various inputs from the user in conjunction with the joystick and button that were added onto the Arduino with an expansion board. The Joystick, being an analog device, has an almost infinite range of values which could be read off of it. Because of this, the onboard Analog to Digital converter was used, which allows the values of the joystick in the program to be read. A multimeter was also used to measure the voltage across the axis of the joystick. Using the voltage values that were measured, the values in the Arduino program were established, meaning that the user has moved the joystick and it needed to respond accordingly. In reading these values, a deadzone on the joystick was also established, which would make sure that any small accidental movements would not change the position of the cursor.

The button control portion of the Arduino code allows the user to select, deselect, and clear icons. To select an icon a user would need to press the button for a short amount of time. The code seeing that the user has pressed the button will check if the icon is able to be selected using the selection array, and if it is the value of the icon in the main array, it will change to reflect that it has been selected. The selection is shown to the user by an overlay being drawn over the icon. To deselect an icon the user must also perform a small press on a selected icon, updating the value and removing the overlay. If the user presses the button for a longer period of time, the program initiates an end of round clear where all full sets of matched icons that are selected are cleared from the screen and all remaining icons are dropped downward to the lowest point possible in the 6x6 display. The blank spots are then re-filled with new randomly generated icons and the selection and data arrays are updated with new correct values.

One of the other portions of code that was needed was a section of automated code that would be used to access the hard drive on the DE10 board. This code, although fairly simple, was necessary in order to decrease the length of the code dramatically. It would be used to access the hard drive in eight bit segments in order to reconstruct the BMP files. As it was mentioned earlier, the hard drive on the board contains a list format of all of the BMP images in eight bit segments so the code is able to take them apart into individual pixel data chunks, so that they can be displayed on the screen.

**Problems Encountered**

The development of this game came with its share of bugs and problems. During the design phase of it, the icons initially created were too dark in appearance and therefore had to be made brighter. The handheld enclosure also went through many phases of designing and 3D printing to ensure that both boards fit nicely inside, and that it was comfortable to use. It went from being a thinner and flat edges open design, to a thicker, softer edges and more closed design.

Furthermore, one of the original problems found when first working on the arduino part of the code was the set up and calibration of the analog joystick component. Since there was no data sheet found on the part, figuring out the X and Y axis was all trial and error. Once the orientation was found and a multimeter was used for calibration of the joystick, it became much less of an issue. Calibration was then only needed a couple more times throughout the project due to imperfect hardware.

In the coding portion of the game there were also quite a few major issues that arose. In the automation section, there was a problem that came up where the bottom row of the BMP images were not being shown correctly. The problem was in the BMP conversion code. When looking at the code, the if statement which processed the data from each BMP was coded to stop at row 31, which resulted in the final row of each BMP being values that were not predictable, resulting in the final row being incorrect when shown on screen. To fix this issue, the 31 in the if statements was changed to a 32, which would then include the final row of the BMP and display it properly.

Lastly, in the main portion of the code an issue was found where data was not being sent properly between the DE10 and Arduino boards. On the first display of the icons on screen, not all icons would be shown at first. Originally this issue was fixed by sending data multiple times to attempt to present all icons. Although this did display all icons, this solution was not a perfect fix since sending the data multiple times resulted in an unintended animation. To fully fix the issue, the delay between sending data had to be increased, allowing the DE10 more time to process the data. However, this solution had the negative side effect of the game becoming slightly less responsive, although this was unavoidable for the overall function of the game.

**Improvement**

Some major improvements that could be made consist of upgrading the overall hardware and circuitry. Eliminating the push button and its circuit could be done by using one of the on board push buttons featured on the Arduino. Adding additional programming versus hardware would be much more efficient, cleaner looking, and provide more functionality. The addition of an external joystick incorporated onto the enclosure could also be very beneficial. It would be more comfortable to hold while also decluttering the top of the arduino board. This would allow for the enclosure to then be slimmed down into a smaller model referring to the overall height of it.

**Conclusion**

In conclusion, although this project has not fully been completed due to some minor bugs that remain in the code, it has come a long way. The project was beneficial for reinforcing previously gained design and programming skills. By completing it, we have been able to expand upon Arduino, DE10 LITE, and Python programming skills, as well as other essential skills such as debugging, pixel art design, 3D design, and FPGA usage. By putting all of these skills to work, the team was able to design and code a Retro gaming system completely from scratch, all with nothing but other games to be inspired by. Thus, this was also done legally without causing harm to any original game designer/creator. Although the game does not work to its full and best extent due to some previously mentioned minor bugs remaining, it is still playable and entertaining.

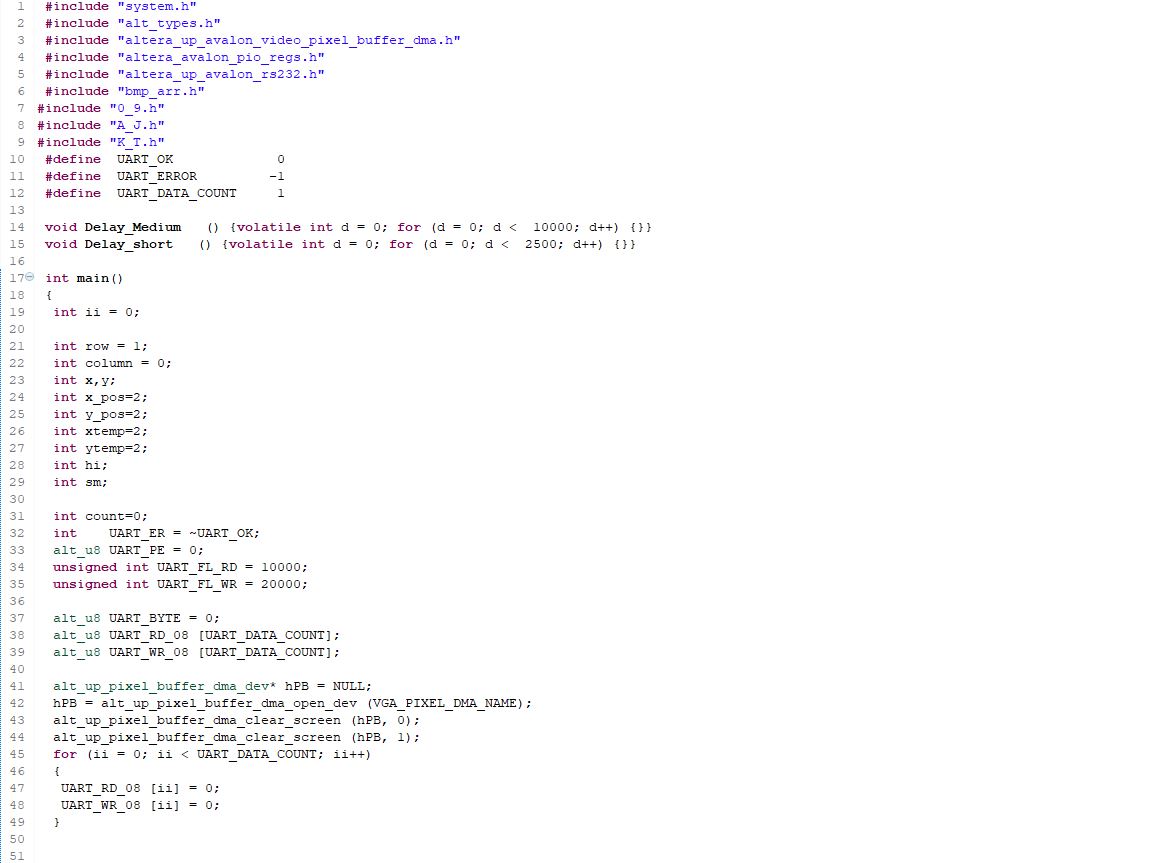
**References**

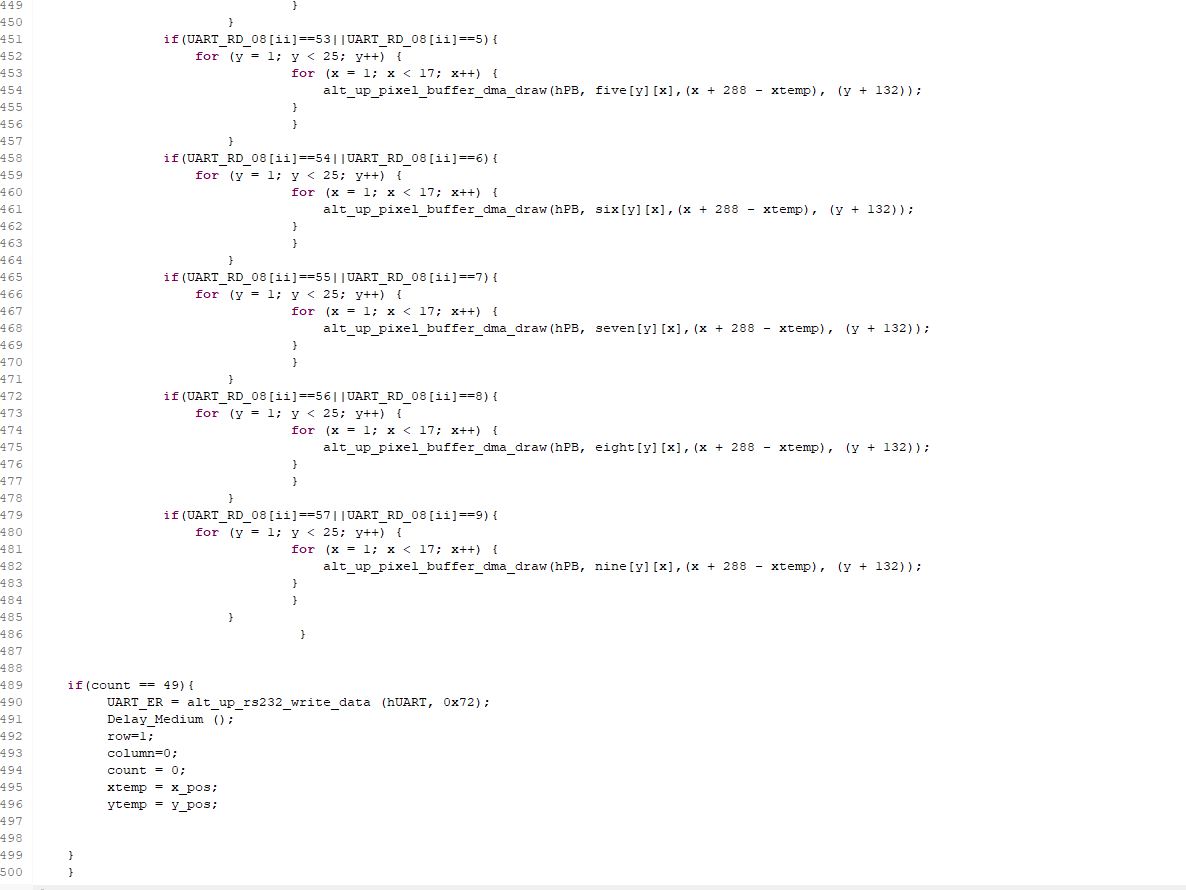
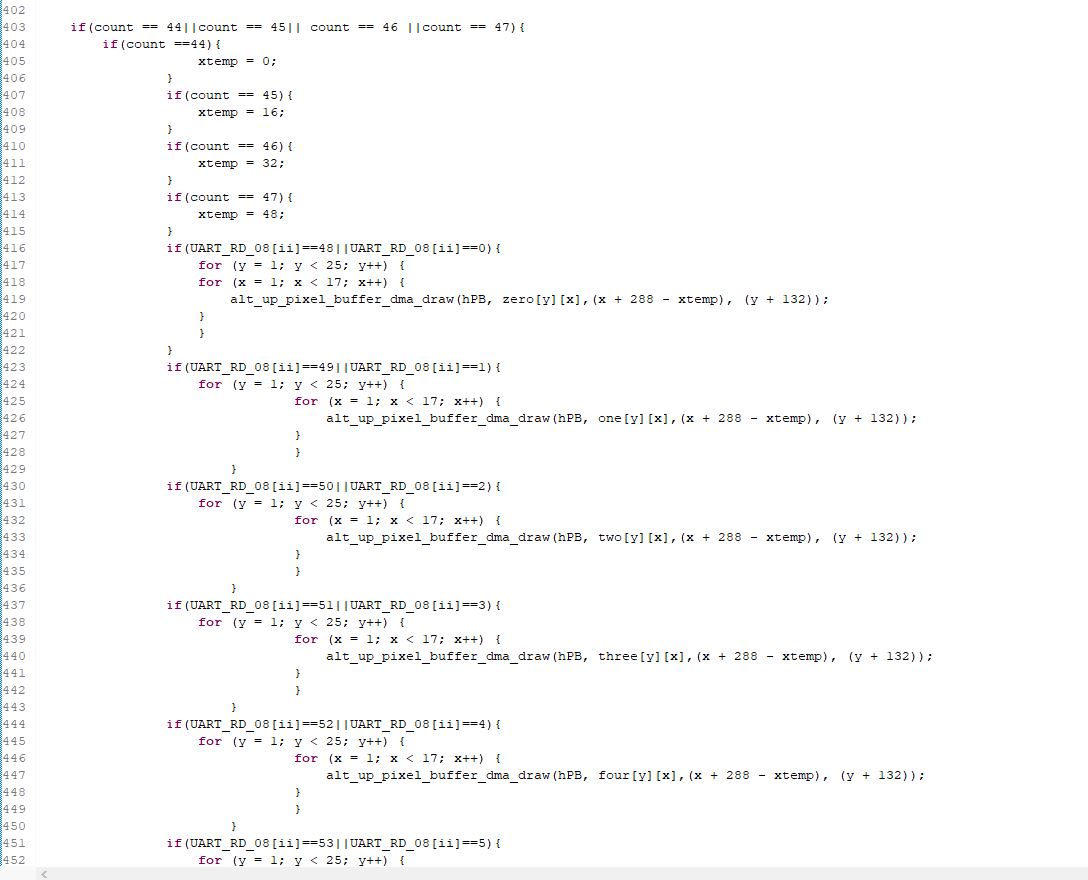
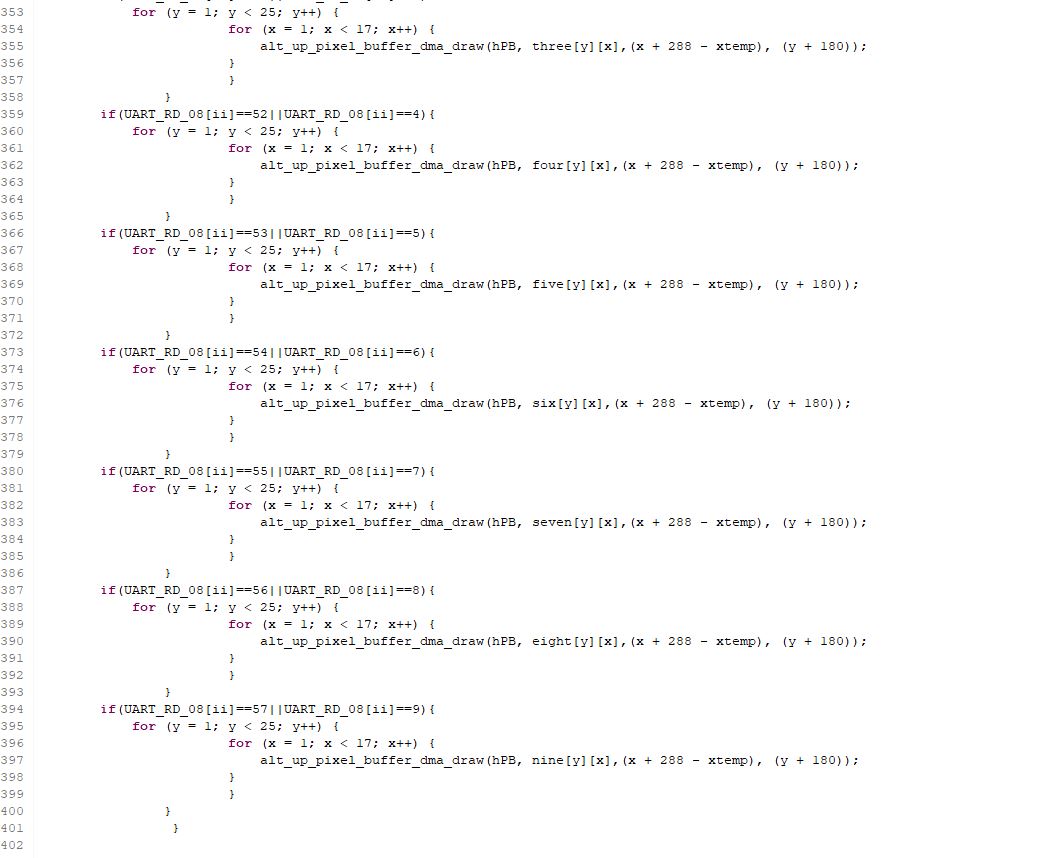
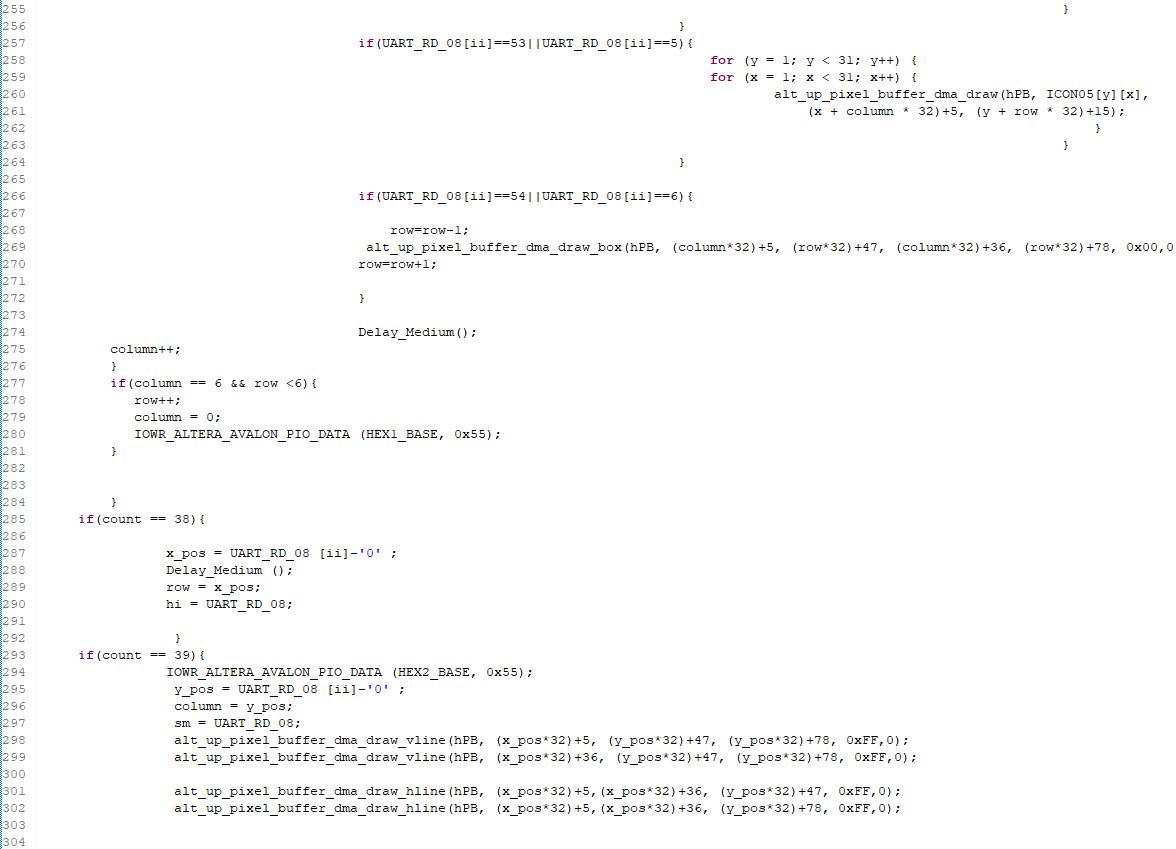
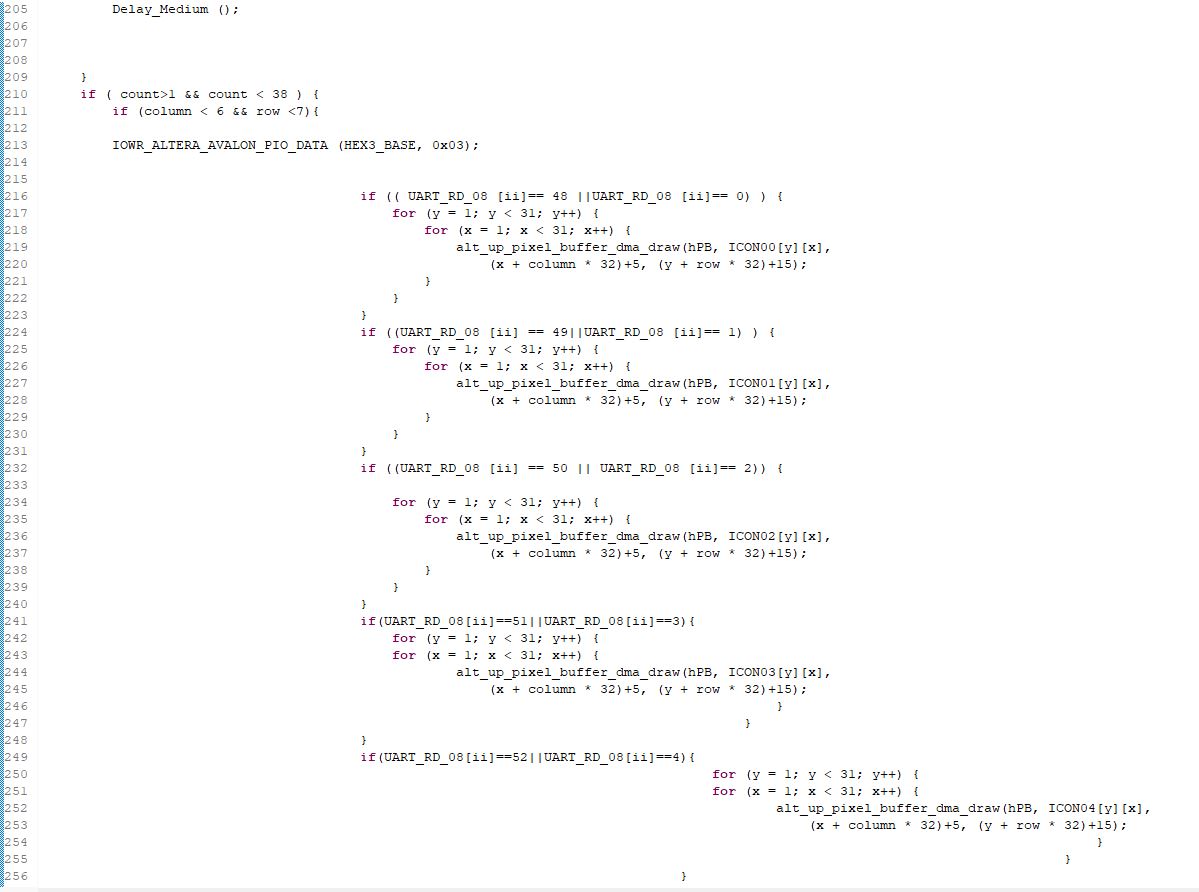
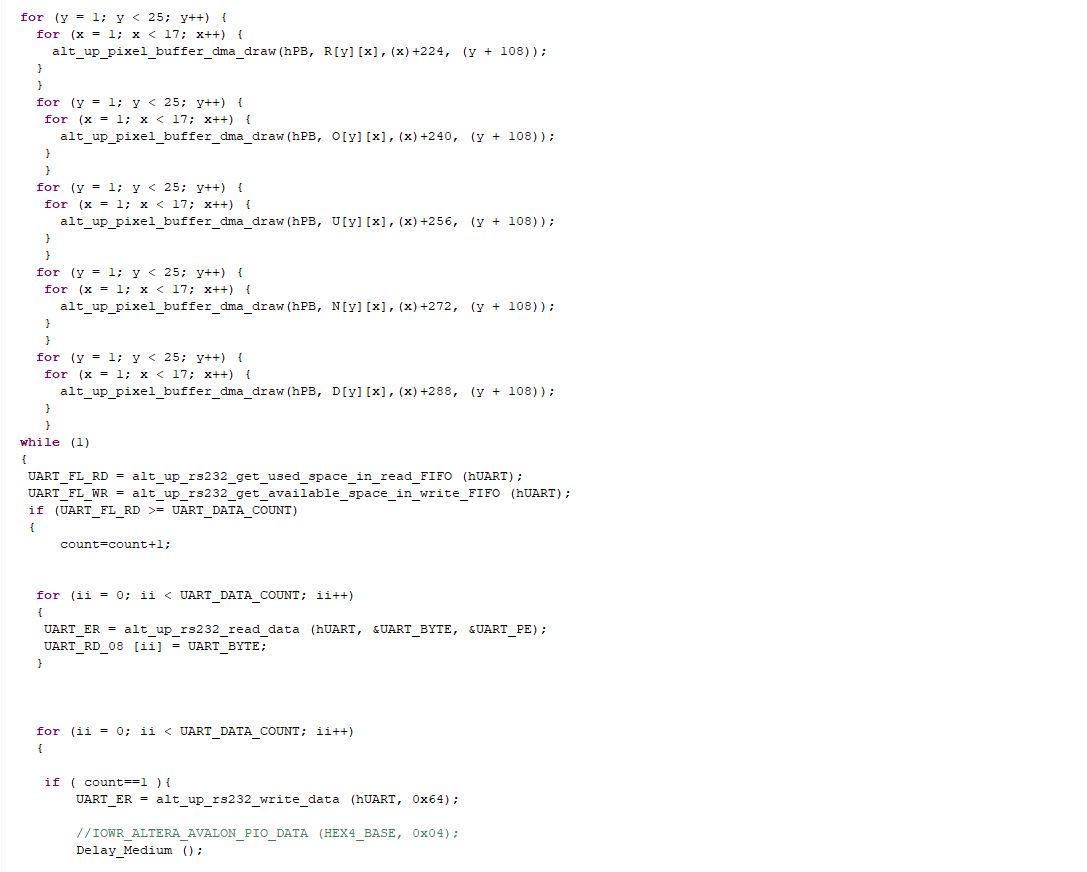
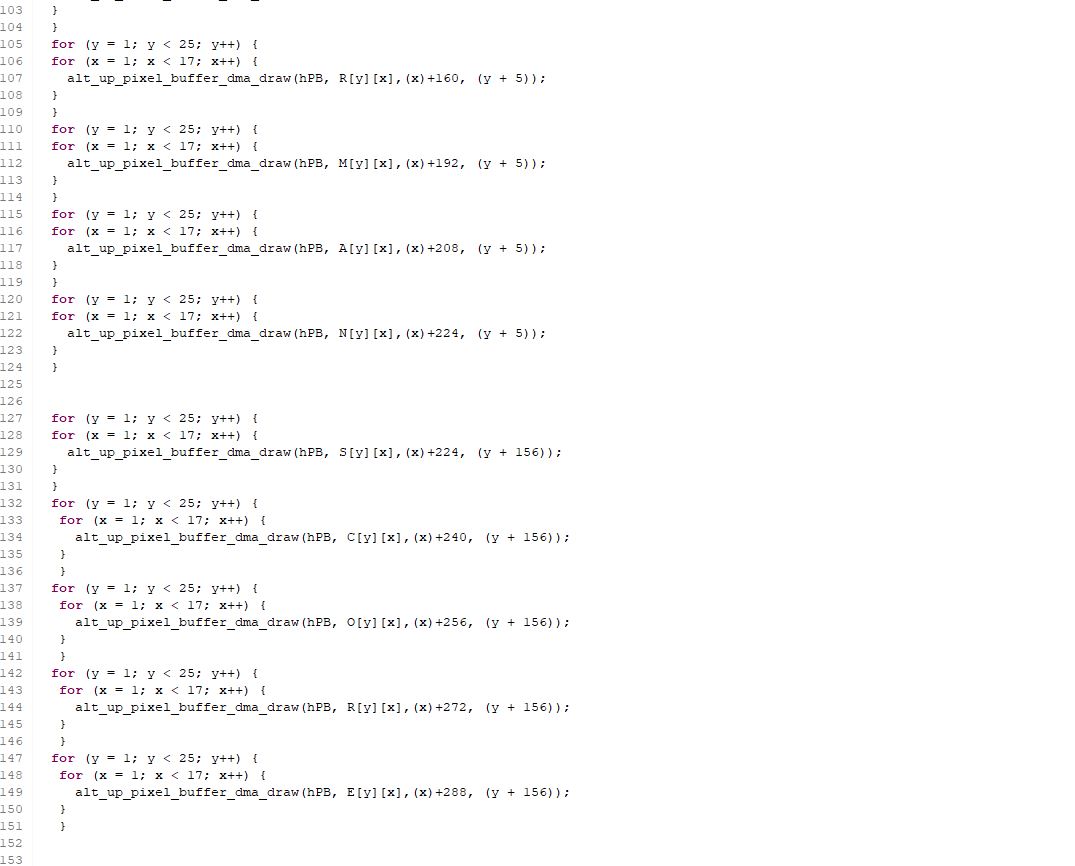
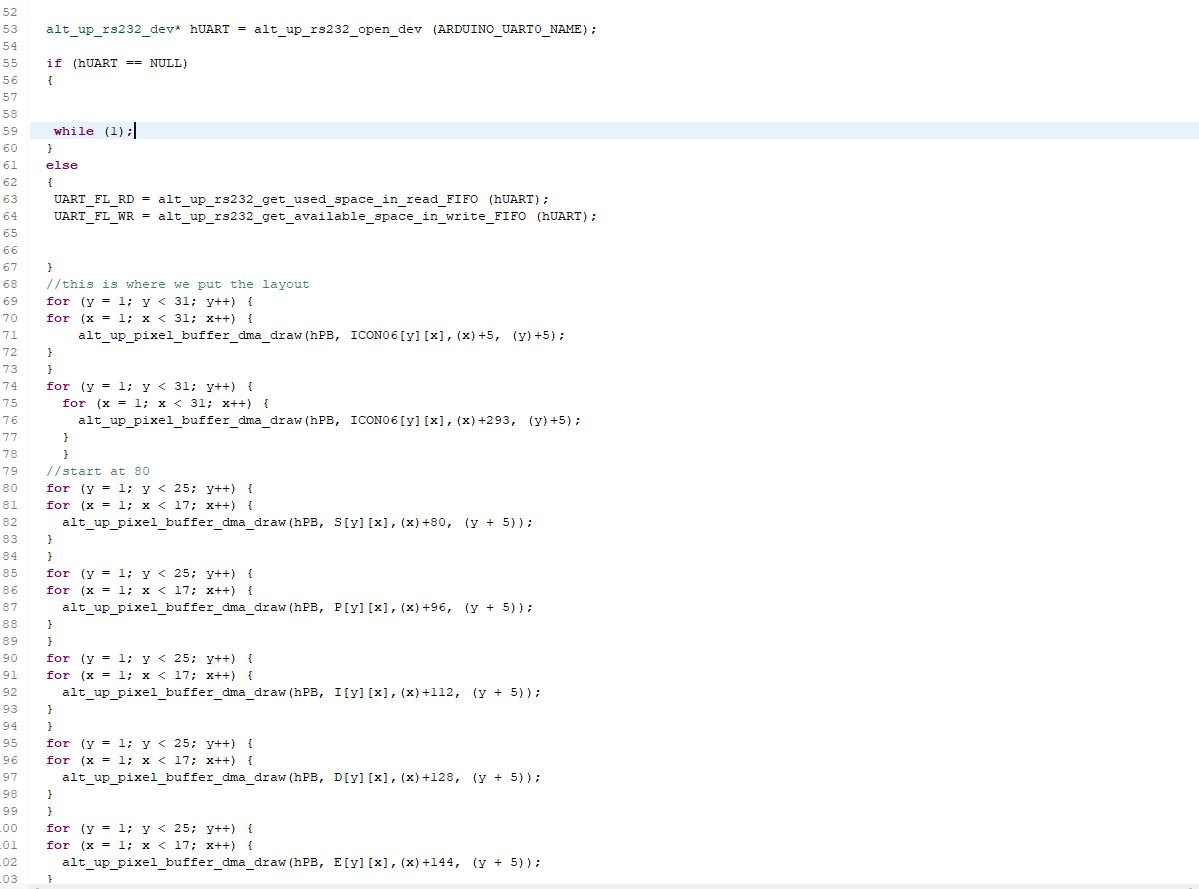
Liesch, N. (n.d.). *THE BMP FILE FORMAT*. The BMP file format.

System. (2013, March 29). *SerialUSB.print() in Output*. Arduino Forum.

**Appendices**

DE10-LITE Code





Arduino Code