Electronic Lego Sorter

Adenike Adeyemi, David Carey, Katrina Little and Nick Steinman

Dept. of Engineering and Computer Science, University of Central Florida, Orlando, Florida, 32816 – 2450

***Abstract* — A high-level engineering project, the Electronic Lego Sorter (ELS) makes use of a number of smaller subsystems that, when working together, achieves the ultimate goal of sorting Legos based on shape and color. With 8 separate buckets to sort into, the ELS allows the user to assign specific pieces to different buckets based upon their own preferences. With a simple, user-friendly touch screen interface, the sorter uses a simple webcam to gather images that are then processed to determine the color and shape of the pieces. This information is then compared to the user's selections of where the pieces should go and sorts them appropriately.**

***Index Terms* — Image processing, Belts, DC motors, microcontrollers,**

I. Introduction

A product of the inconvenience of hand sorting Legos, the ELS was conceived as a device designed to save time for the user. In the case where a builder doesn't finish a current Lego project, storing the unused Legos can be messy and tedious, doing nothing more than causing frustration and wasting time. An Arduino microcontroller will be the main control unit for the system, handling the various mechanical parts, as well as also handling the touch-screen user interface. The image processing will be performed by a simple Logitech C110 USB webcam connected to a BeagleBone Black. After the user interface gathers the details on the assignments for the Lego receptacles, the lift system will then lift small amounts of Legos in specific intervals onto the conveyor system. The first conveyor will move at very low speeds in order to dispense the Legos one at a time onto the lower conveyor which will be moving faster to create space between the Legos. With the Legos passing underneath it one at a time, the webcam is able to take pictures in intervals to determine when a Lego is completely on screen, at which point the system will perform a full analysis of the piece gathering the necessary dimensions and color values that will determine the piece's placement in the receptacles. Finally, a sweeper component will be used to push the Lego from the conveyor belt, down into the rotating arm which will guide the Lego into the proper receptacle. This process will repeat until Legos are no longer passing underneath the camera, or the user forces a shut down.

II. Subsystems

The ELS as a whole is simply a sum of its parts. The best way to describe the components of this project is to explain how each of the subsystems work individually in order to perform the ultimate goal. This section will be a mild technical explanation of each of the subsystems.

A. User Interface

The user interface is designed to be easily accessible and understandable to users of all ages. The simplest way to do that is to use tools that are very common for user interfaces in other technologies. For these reasons, it was decided that a touch screen would be used. Specifically the RA8875 TFT Resistive Touch Screen. With the extensive documentation provided by the manufacturer, it was fairly simple to get the screen to work with the microcontroller. The user interface will allow users to choose what types of Legos each receptacle will hold. After assigning each of the 8 receptacles the Legos that the user wants, there is a final confirmation screen before the system begins to start working.

B. Lift System

The lift system, simply put, is a box that is raised a lowered along a threaded rod to push the Legos onto the conveyor system. The box is built out of a very light, but sturdy ¼ inch plywood held together by screwing the edges into a small 1x2x4 inch section of wood. The Legos will rest on the top of the box as they are being lifted, and to keep the Legos from falling off the lift, the sides are extended 4 inches past the top of the lift. The bottom section acts as the point where the box is being lifted. A small hole is drilled in the center where a coupler nut is epoxied. This nut moves up and down a threaded rod which is rotated by a (insert linear actuator part here). The box is also attached to a longer base using to drawer slides to increase the efficiency of the lift, reduce the load on the motor, and provide the necessary backing to keep Legos from falling off the lift.

C. Conveyor System

The dual conveyor system will be used as the means to separate Legos so that the camera only has to process one Lego at a time. With photograph paper as the material for the belt and a 12V DC motor for each, the first belt will move slowly enough that only one Lego at a time will drop onto the lower belt. With the lower belt moving at a slightly faster right, it will allow each Lego to get some space from each other, making it so that the image processing chamber will only have to process one Lego at a time.

D. Image Processing Chamber

The Image Processing Chamber (IPC) is made up of three components. To take the pictures, a simple Logitech C110 USB webcam is used, due to its cheap price point. In selecting a camera, it was taken into account that it would only need to analyze simple shapes and colors, so a more elaborate, more expensive camera would be superfluous. The camera is paired together with a small mirror set at a 45° angle. This allows the camera to get both a top view and a side view of the Lego. To perform the processing for the system, the webcam is connected to a BeagleBone Black computer. The camera will be converting images to JPG files which the BeagleBone Black will process by simply gathering the color codes from the array. With a single-color background on the conveyor underneath, it will be incredibly simple for the program to differentiate between the Legos and the background. This same method will also be used to gather the dimensions of the Lego to determine its shape.

E. Sweeper

The IPC will be set up directly overhead the sweeper mechanism. The conveyor system will drop each Lego into this area, and once the image has been processed, and the rotating arm has been properly repositioned, a simple 12V DC motor will push the sweeper in order to send the Lego off the edge, directly into the rotating arm system.

F. Rotating Arm

The rotating arm is the final means of transportation for the Legos. The rotating arm simply consists of a slide attached to a 5V stepper motor. With buckets of Legos waiting below, the arm will get the information it needs from the Arduino microcontroller to determine which bucket to rotate to.

III. Hardware Design Detail

The following section goes into detail on the hardware used to design each of the specific subsystems in the ELS, as well as the power supply. While they all ultimately come together to make the final product, each subsystem has very specific needs that need to be met individually, through a series of motors and parts.

A. Power Supply

Since this sorter is intended for in home use, the power supply will be stepped down from a standard AC 120V 60 Hz wall outlet. The subsystem components operate at 12V DC as well as 5V DC. To allow for continuous operation 20% current will be added to the rated current as shown in Table I.

Table 1

POWER RATING		
	Supply 1	**Supply 2**
Rated Voltage [V]	12	5
Rated Current [A]	1.42	2.84
Rated Current +20% [A]	1.7	3.41
Power Consumed [W]	20.4	17.05

The two supply voltages needed for the project were designed using T.I.'s power architect program Webench. Supply 1 is a wide input range boost DC/DC converter. Supply 2 is a 5A synchronous step down DCAP2 mode converter. After inputting the specifications of the two load supply voltages the source input voltage to the two converters was calculated to be 12V, 3.327A as shown in Fig 1. The AC/DC portion of the power supply was purchased from Jameco electronics to supply the two converters and is rated at 12V, 3.7A.

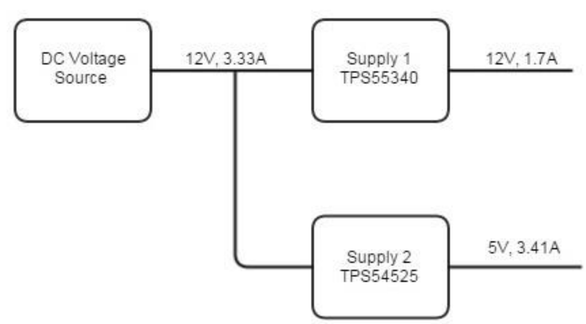


Fig. 1. Block diagram created using Webench Power Architect.

The PCB board for the power supply was designed using EagleCad. Both the 12V and 5V DC loads were designed on one board using 2 layers. The AC/DC converter will be connected to the board and supply the

TABLE II
COMPONENT POWER CONSUMPTION

Component	Rated Voltage [V]	Rated Current [A]	Power Consumed [W]
Supply 1			
Lift Arm DC Motor	12	0.3	3.6
Conveyor Belt DC Motor #1	12	0.3	3.6
Conveyor Belt DC Motor #2	12	0.3	3.6
Atmega32u4	12	0.52	6.24
Total		1.42	17.04
Supply 2			
Lift Arm Microswitch #1	5	0.17	0.85
Lift Arm Microswitch #2	5	0.17	0.85
Lift Arm Controller (L293D)	5	1.2	6
Rotating Arm Stepper Motor	5	0.32	1.6
Rotating Arm Controller (UNL3003)	5	0.5	2.5
Rotating Arm Photoelectric Sensor	5	0.02	0.1
Beaglebone Black	5	0.46	2.3
Total		2.84	14.2

12V, 3.3A input. The board was laid out very carefully to draw multiple polygons. Aside from ground, there were three additional polygons drawn to improve signal strength

and allow for more current flow. Besides ground, the other three polygons were V_{in}, V_{O1}, and V_{O2}. The board was laid out in such a way that only components connected to these voltages were in that particular polygon space.

A DPDT kill switch will be attached to the AC supply voltage to shut off the power to the sorter when not in use.

B. Lift System

The lift system is the first stage of the sorting process. The lift system will be propped up against the first conveyor belt. A bucket will be placed right next to the lift system. The bucket will be where the user drops their unsorted LEGO parts. The lift system will be constructed using a threaded rod and will have a "traveling nut." The nut will be fixed to a platform that moves up and down which is guided by 6" drawer racks.

As the motor turns CW and CCW, the platform will move up or down. To begin, this platform will be level with the dump bucket. The bucket will be positioned at a slight angle to the lift system. To start, the moving platform will drop below the dump bucket 6" allowing parts to fall into the crevice. The platform will move downwards until it hits a mechanical micro switch. This will allow a good amount of LEGO parts to fall onto the platform. When the switch is triggered, it will send a signal to the microcontroller to stop the motor and reverse direction using interrupts. The platform will now move upward until it hits another limit switch. When the upper limit switch is triggered the platform will now be elevated to the height of the conveyor belt and the parts will spill out onto the belt, and the process will start over.

There were three types of motors that were initially considered for the lift system: Stepper motors, linear actuators, and DC motors. Stepper motors would eliminate the need of feedback sensors and would be the easiest to control. The stepper motor was ultimately ruled out because they typically only operate at very low RPM. Linear actuators seem ideal because of their "all in one" construction with built in limit switches. Using a Linear actuator would make the mechanical construction a lot easier to build but they are very pricey. The motion of travel would also be limited because linear actuators only come in certain sizes. Finally, the RPM of a linear actuator is not as low as a stepper motor, but it also does not go as high as a simple DC motor. It was ultimately decided that a DC motor would be the best choice coupled with an upper and lower limit switch.

The lift system needs to move up and down in no longer than 10s. In order to do this a high RPM DC motor was selected. The threaded rod that was coupled to the motor had 23 [rev/in]. A 5500 RPM motor was selected that operates at 12V DC. The motor was predicted to perform 1 iteration of moving up and down in about 3s. In testing it was probably more like 5s due to mechanical load but was still a little too fast. The motor's speed was scaled down using the PWM pins of the Arduino Leonardo.

The motor is to be controlled by an L293D H-bridge. The H-bridge circuit changes the motor direction by turning the internal transistors on and off. This causes the current flow through the H-bridge to change, which causes the motor to reverse direction. Figure 2 below shows a circuit diagram of the lift system motor.

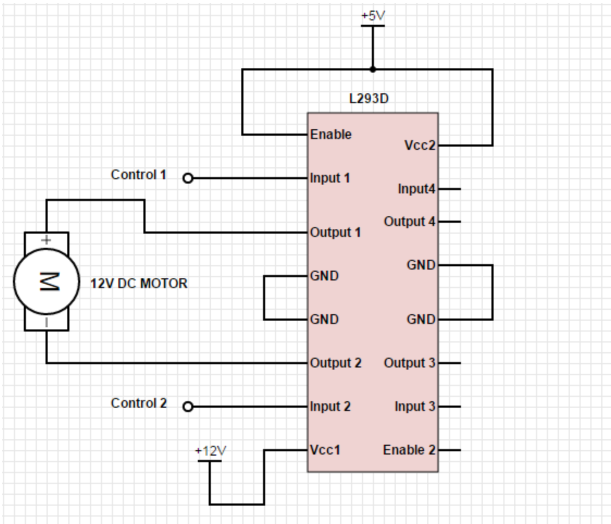


Fig. 2. Lift arm motor driver circuit.

When the input "Control 1" from the MCU is held high and "Control 2" is held low, the motor turns in one direction. When "Control 1" is held low and "Control 2" is high, the motor spins in the opposite direction.

C. Conveyor System

The conveyor belt motors are geared, high torque, and operate at 12V in this project. The conveyor system moves in one direction only, so use of more complex bi-directional circuitry is not required. The motors are isolated from the lower current microcontroller through TIP120 darling pair transistor packages. The electrical connection diagram of a motor is shown below in Figure 3.

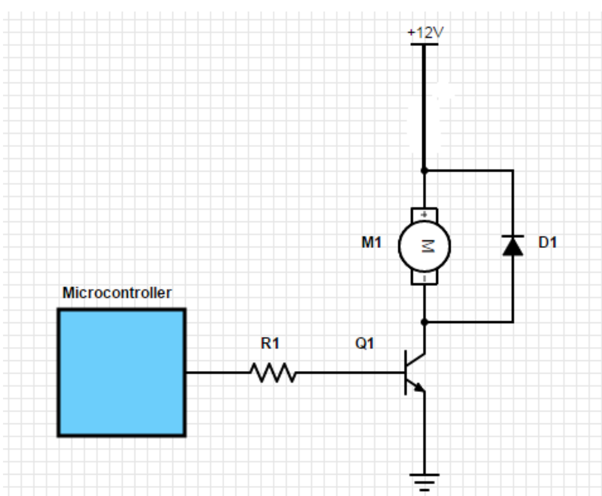


Fig. 3. Motor driver circuit for the conveyor belt motors.

The device Q1 is a Darlington transistor pair. The package has a high gain and only requires a very small input base current. The MCU can only handle 40mA per pin, thus having a high gain from the transistor package is essential. Diode D1 is a flyback diode to minimize voltage and reverse current spikes from motor activation. Resistor R1 is a current limiting resistor which also biases the transistor package into saturation mode. This circuit is used in conjunction with pulse width modulation control from the MCU. The PWM control allows for varying the speeds of the conveyor belts. The top belt moves very slowly, while the bottom belt moves at a higher rate of speed. This is done to allow for crowded LEGO pieces on the top belt to separate from each other when they fall onto the bottom belt. The belts will have their speeds vary as well as turn on and off to allow for image processing.

D. Sweeper

The sweeper system is to be positioned below the second conveyor belt. The sweeper has been built from LEGO technic parts. Each side of the sweeper is lined with gear racks. An axle with gears attached on each side will drive the gear racks back and forth. A low RPM DC motor will be coupled to the axle that connects the two gears. The sweeper will be implemented in a very similar fashion to the lift system. As with the lift system, there will be two limit switches. One switch will be placed below the second belt, behind the sweeping bar and the other will be placed on the edge of the table that the sorter sits on. The motor direction will be controlled by an L293D H-bridge controller. Once a LEGO part has been identified by the image processing chamber, the motor will be turned on. The sweeper arm will push the LEGO part to the edge of the table until it hits the second limit switch. It will then reverse direction until it hits the switch underneath the belt. At which time it will stop and wait for the next LEGO part to be processed. The sweeper is shown below in Figure 4.

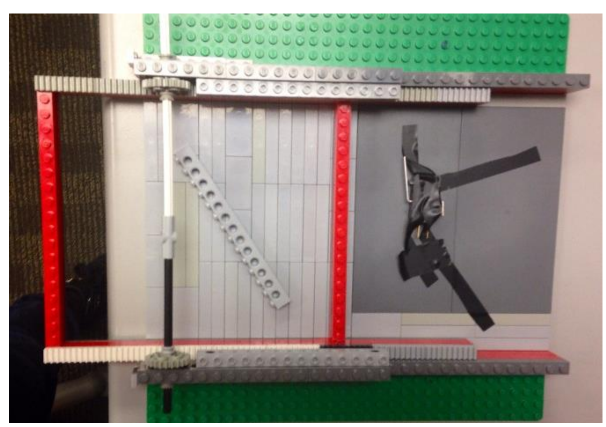


Fig. 4. The sweeper will push Legos from the image processing chamber down into the rotating arm.

E. Rotating Arm

The rotating arm's task is to position itself over the proper bin to allow the identified LEGO piece to fall in. Specifications for the rotating arm include the ability to rotate 360 degrees and have an accuracy of at least 10 degrees. Stepper motors offer 360 degree continuous rotation as well as incremental and accurate stepping. Because the rotating arm shaft is light-weight and can be coupled directly to the motor, a high voltage, high torque motor is not necessary. As such, the small 5V rated 28BYJ-48 stepper motor is used to move the rotating arm.

While stepper motors can be very accurate, they lack sense of relative positioning. With extended use the rotating arm will become more and more inaccurate, straying further and further from the correct positions. A simple but effective solution has been developed to navigate this issue. A sensor must be used to provide positioning feedback to the system. The method used in this project utilizes an infrared reflectance sensor and a white disk with an IR absorbing line as shown in Figure 5 below.

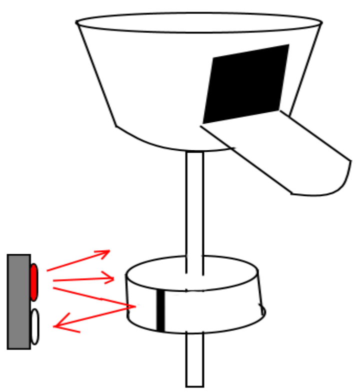


Fig. 5. The rotating arm uses a sensor to reposition itself to accommodate for a loss of accuracy over time.

The IR absorbing line on the white disk provides an identifying point or "home" position for the sensor to read. To home the rotational arm, the arm is rotated and a polling loop is ran on the sensor until the home line is read. When the sensor outputs a significantly smaller analog value from the black line, the rotational arm is at home position. From this point, the MCU can calculate and send the stepper motor the number of steps required to get from bin to bin. The stepper motor's gear ratio is approximately 64:1 on full step mode. Because the exact ratio is not an integer, the motor loses accuracy over time. This is a relevant consideration for this project as the sorter is to be left on for extended periods of time. To mitigate this issue, the motor is sent to home position periodically between sorting pieces.

Sorting time is also an issue for this project. The movement of the rotating arm is optimized to ensure the arm never moves more than half way around when moving from bin to bin. The code calculates the two distances between the current location of the arm and the target bin. Figure 4 below describes visually these two distances.

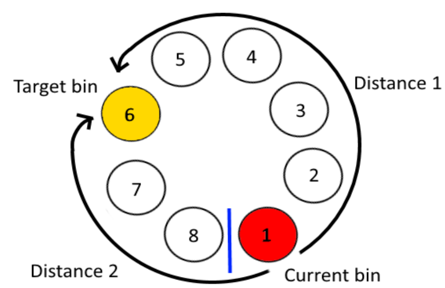


Fig. 6. The rotating arm will calculate the shortest distance between the current location and the destination.

With an 8 bin configuration, the maximum distance the rotating arm will move during any bin to bin movement is 4 bins. In Figure 3 above, the arm will take the path "Distance 2" because it is the shorter path. The distances are calculated with the following two equations:

$$\text{Distance1} = \text{bigger} - \text{smaller} \quad (1)$$

$$\text{Distance2} = (\text{number\_of\_bins} - \text{bigger}) + \text{smaller} \quad (2)$$

In the equations above, the term "bigger" refers to the higher bin number of either the current bin or target bin. Likewise, the term "smaller" refers to the smaller of the two bins. The algorithm calculates both distances, and through a series of nested if statements, it determines which direction it should rotate. The determined bin distance is then converted from bin numbers to number of steps the motor must take to reach the target using equation (3) shown below.

$$\text{Number\_of\_steps} = \text{Distance}*(512/\text{number\_of\_bins}) \quad (3)$$

The 512 term comes from the number of steps it takes the stepper motor to make a full revolution using a half step coil energizing sequence. With these equations the motor will be able to do the necessary job that must be done, while putting the least amount of strain on the motor.

The stepper motor draws too much current for the MCU to handle, so a ULN2003 Darlington transistor package is used in between the motor and MCU. The stepper motor uses 4 control wires to energize its coils in a specific sequence as well as a 5V power line. A circuit diagram of the stepper motor configuration is shown below in Figure 7.

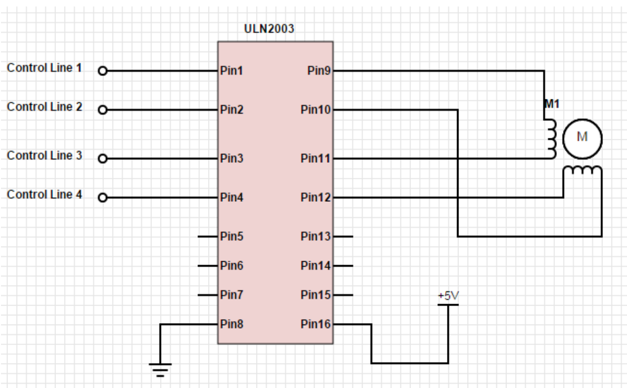


Fig. 7. The driver circuit for the rotating arm stepper motor.

F. User Interface

The screen being used in this project is the RA8875 RAiO touch screen display as it fulfilled our requirements. This LCD screen will be connected to the Atmega32u4, which will handle controlling the LCD screen and touch screen interrupts. For communication 4-wire SPI serial interface is used for the LCD and Leonardo. Using two wires connected from the display's parallel interface ports to the Atmega32u4's digital pins will control the touch screen and that will controls the interrupt and reset. For power it is configured to run at 5 volts.

G. Image Processing Chamber

The Image Processing Chamber has fairly minimal hardware involved. The two major components of the IPC are the BeagleBone Black and the Logitech C110 USB webcam. The BeagleBone Black is being used specifically for its processing power and memory. It has the necessary power to be able to run the webcam and process the images that it receives fairly quickly. This makes it an ideal choice for the Lego sorter.

H. Master Control Unit

The MCU used in the project is an Atmega32U4 by Atmel. The Atmega32U4 has 32kB of flash code memory and 2.5 kB of dynamic memory. The MCU is the central processing unit and handles all of the sensor inputs, motor control, LCD and Beaglebone Black communication. Table 3 below shows the devices used and the required pin functionality from the MCU.

Device	Pin Functionality	# of pins
Rotating Arm Stepper Motor	Digital I/O	4
IR Reflectance Sensor	ADC	1
Conveyor Motor 1	PWM	1
Conveyor Motor 2	PWM	1
Lift System Motor	PWM	2
Lift System Switch Sensor	Digital I/O	1
Lift System Switch Sensor	Digital I/O	1
Beaglebone Black	SPI	4 (1 unique)
LCD Touchscreen	SPI	4 (1 unique)

TABLE 3

Under pin functionality, "ADC" refers to the on-chip analog to digital converter of the Atmega32U4. The sensor using the ADC pin sends an analog signal that the MCU converts to a 10 bit digital value between 0 and 512. The abbreviation "PWM" refers to the on-chip pulse width modulation channels that can be connected. The Beaglebone Black and LCD touchscreen interface with the MCU via an SPI master slave configuration. The wiring diagram is shown below in Figure 8.

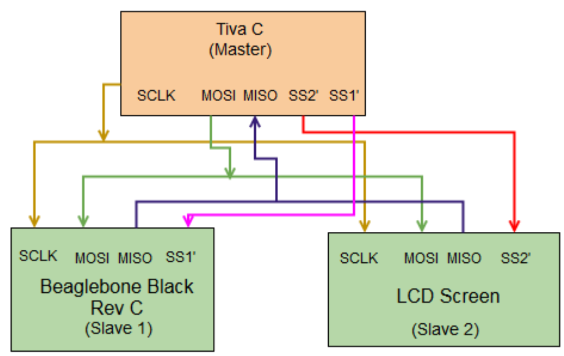


Fig. 8. Master/slave diagram between the MCU, Beaglebone Black, and LCD screen.

The devices all share the same SCLK, MOSI, and MISO lines. Communication is done by the master device selecting a slave device to send data to or receive data from using the slave select lines. After the LCD screen finishes its initial setup, most of the communication will come from the Beaglebone Black sending data about each processed LEGO piece. The MCU along with all of the supporting device circuitry is on a PCB and was designed using Eagle PCB software.

IV. Software Design Detail

The software involved with this project is what keeps it organized. The software determines specifically what parts are running at what time and for how long. This section will outline the specifics of the software design.

A. User Interface

The User Interface for the system is designed with simplicity as the main focus. This was the main reason a touch screen was considered in this project. A resistive touch screen was chosen because of its low price compared to a capacitor touch screen and less complexity. Figure 8 shows a flow chart for the User Interface.

The screen will turn on and display the system title, version number and a start button.

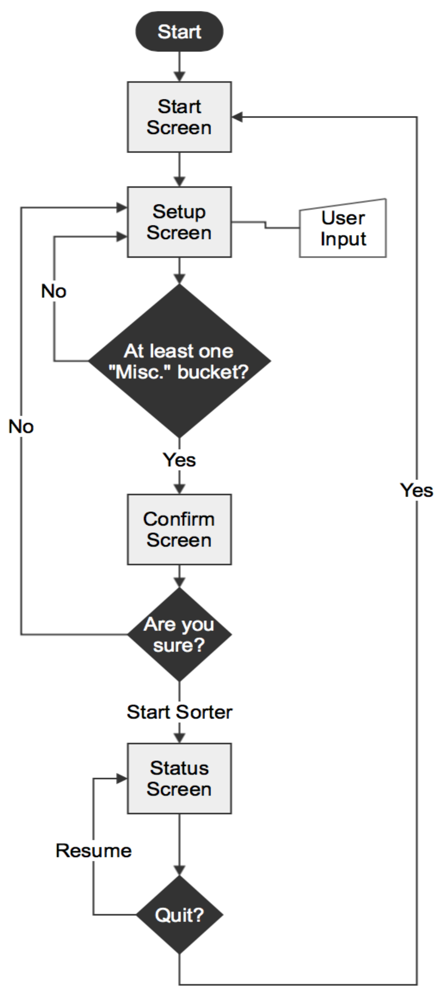


Fig. 8. The flow chart for the user interface.

When the start button is pressed the User will be taken to the Selection screen. Here the User will decide how to sort the Lego (Shape or Color) and assign which bucket will contain which part. By default all the buckets are assigned with “Misc.” which mean that it will act as an error bucket that will hold any blocks that cannot be determined by the image processor. By selecting the bucket number. The bucket number is highlighted and a window appears beside it. The User will shift through the window for the color or shape will be assigned to that bucket and double that choice to confirm. The User is not allowed to move to the next page unless at least one bucket is reserved as a “Misc.” otherwise a warning window will appear notifying the User. Once prepared the User can press the Next button on the bottom right corner.

The next window is a confirmation page where the user will see a list of what item is assigned to what bucket. Here the user can decide to either go back to reassign what is to be sorted by pressing the previous button on the lower left corner or start the sorting process by pressing next.

The last window is a page that notifies the User that the sorter is running and a pause button. Here the user has the ability to pause the sorter for whatever reason. When paused, a window will appear giving the user to resume the system job or exit this job stopping the entire system completely. When stopped the User will be brought back to the starting window.

Implementation of the touchscreen was a challenge as the RA8875 LCD was overly sensitive to any sort of pressure, and as a result it would register false positives, or “ghost touches.” In order to counteract the problem, the system will turn off touch interrupt long enough for the user to lift their finger and zero out the touch data. This eliminates multiple touches occurring in sequence as a result of the ghost touches.

B. Image Processing

The image processing component of the Lego sorter primarily makes use of the built-in JPG library. The software starts by gathering images from the webcam in intervals of .5 seconds. With a solid color background, analyzing the images to find changes in color becomes a very simple task. All the program has to do is look for colors that have changed to a degree that could only be explained by an object on screen. For example, if the value of a pixel in the picture changes by only 1 degree from one picture to the next, then the picture has likely only had a small change in lighting and can likely be ignored. However if a large area of the picture has suddenly changed by 100, this must be a Lego that has arrived in the chamber.

At this point the software is analyzing a few select features. First and foremost it determines the color of the Lego. This is simply done by gathering the color code from the pixels. The slightly trickier feature is the shape. Due to the unique nature of each and every piece, the pieces are distinguished with a few small measurements. The overall surface area is determined by the top view. The side view will gather the RANGE of heights of the piece. Using these two dimensions, they will be compared to data that has been previously hard coded to determine the shape of the Lego.

C. Overall Organization

The other subsystems don't have as much detail in their software, other than the fact that they need to be kept to a specific timeline. For this reason, all of these components are monitored by the Leonardo microcontroller.

The lift system will simply need to be sent a signal to know when to lift. This signal will be sent in intervals that are dependent on the speed of the first conveyor. To keep the conveyor belts from overflowing, these intervals will be separated so as to give the belt enough time to move the current Legos out of the way.

The conveyor belts will be moving constantly, at constant speeds. After receiving a signal that the user interface is ready, the conveyors will immediately begin, and run until termination.

The rotating arm will move after receiving the necessary data destination from the MCU. After processing the image, the BeagleBone will send the results back to the MCU, which will then compare the results to the destinations the users have prepared, allowing it to determine the destination of the Lego. This destination is then sent to the rotating arm which will adjust itself accordingly.

Finally, the sweeper will wait until after the rotating arm has performed its task before send the Lego into the rotating arm and, ultimately, into its final destination.

V. CONCLUSION

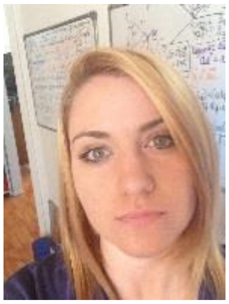
The project that has been put together has been a challenging a rewarding experience. It has pushed the engineers to reach out and learn more than what they've simply learned in classes and has provided an excellent way to discover a practical application for the skills they've gathered.

VI. ACKNOWLEDGMENTS

The group would like to thank and acknowledge Dr. Samuel Richie for his patience and guidance, particularly in the early stages of the project.

VII. THE ENGINEERS

David Carey is a Computer Engineering student. He has recently accepted a position as a Software Engineer at Northrop Grumman and will be residing in Melbourne upon the completion of his degree.



Katrina Little is an Electrical Engineering student. After graduation she hopes to move out of state and pursue a career in semi-conductor research or designing amplifiers.

Nick Steinman is an Electrical Engineering student. After graduation he hopes to pursue a career in embedded system design.

Adenike Adeyemi is a Computer Engineering student. After graduation she hopes to find a career in software development. Specifically she is interested in AI development or interface design.