AC 2011-832: AUTOMATED REMOTE-CONTROLLED DRINK DISPENSING SYSTEM FOR THE PHYSICALLY IMPAIRED: A CAPSTONE PROJECT

Antonio Soares, Florida A&M Univeristy

Antonio Soares was born in Luanda, Angola, in 1972. He received a Bachelor of Science degree in Electrical Engineering from Florida Agricultural and Mechanical University in Tallahassee, Florida in December 1998. He continued his education by obtaining a Master of Science degree in Electrical Engineering from Florida Agricultural and Mechanical University in December of 2000 with focus on semiconductor devices, semiconductor physics, Optoelectronics and Integrated Circuit Design. Antonio then worked for Medtronic as a full-time Integrated Circuit Designer until November 2003. Antonio started his pursuit of the Doctor of Philosophy degree at the Florida Agricultural and Mechanical University in January 2004 under the supervision of Dr. Reginald Perry. Upon completion of his PhD, Dr. Soares was immediately hired as an assistant professor (Tenure Track) in the Electronic Engineering Technology department at FAMU. Dr. Soares has made many contributions to the department, from curriculum improvements, to ABET accreditation, and more recently by securing a grant with the department of education for more than half a million dollars.

Chao Li, Florida A&M University/Florida State University

Dr. Chao Li is currently working at Florida A&M University as an assistant professor in Electronic Engineering Technology. He is teaching Electronic and Computer Engineering Technology Courses. He obtained his BSEE degree from Xi'an Jiaotong University and MSEE degree from University of Electronic Science and Technology of China. He received his PHD in EE from Florida International University. He is an IEEE Member and a Member in ASEE. His research interests include signal processing, biometrics, embedded microcontroller design, application of new instructional technology in classroom instruction.

Automated Remote-Controlled Drink Dispensing System for the Physically Impaired: A Capstone Project

Abstract

The continuous societal transformations around the world are placing considerable strains in the way people are living today. Life seems to be on a clock and changes in technology are being driven by such fast pace in our daily existence. It is already apparent that the human warmth is fading away in many cultures and for people requiring extra human interaction, life is becoming increasingly challenging. It is even more challenging for people with disabilities. Researchers around the world are working diligently to improve the lives of those affected by illnesses that render them disabled. Despite all the efforts, many people are still confined to restricted locations and positions for long periods of time and with little or no interaction with others. In such cases, research in the assisted living area, such as bathing, grooming, dressing, eating drinking and more, entails more considerations. Robots and automated systems is an area that seems to be promising to accomplish some of these tasks.

In this work, the design of a fully automated remote controlled drink dispenser aimed to assist disabled individual is presented. The system is designed to assist those with limited mobility quench their thirst, with the push a button on a typical universal television remote control, the same control used to control household electronics. A robot is activated and proceeds to a drink holding unit. The drink is then dispensed to the strategically parked robot which will then atomically return to the original location where the user can retrieve the drink. The user can choose from three beverage choices by pressing 1, 2, or 3 on the remote.

Introduction

The Electronic Engineering Technology Program at our university implemented several years ago a capstone course intended to capture graduating seniors' ability to design and implement a complete project using design methods and tools used in today's industry. It is a two semester course designed to capture many aspects of engineering and to allow the students to integrate the knowledge received over the years in the program. The Senior Design Proposal (EET 4950) is offered during the fall and the Senior Technical Design Project (EET 4914) is offered during the spring semester. During the proposal, the student must introduce an innovative problem, devise techniques to solve such problem, generate cost analysis, and present time frame for the deliverables. Deliverables include research findings, presentation, laboratory notebook and technical report. During the design portion, the student must analyze the problem, use design software tools to verify design, fabricate PCB boards from design files (if applicable), implement the design, maintain a lab notebook, make a final presentation with demonstration and write the final technical report. The student final project and presentation are judged by all faculty members in the department and additional guests from other departments and industry.

In this project, the student combines technology and his personal charitable drive to design a system aimed to assist the physically impaired. The Coolerbot Project is composed of three main sections: (1) the Coolerbot, (2) the Cooler Station, and (3) the Controls. The system's operation is depicted in the block diagram of Figure 1.

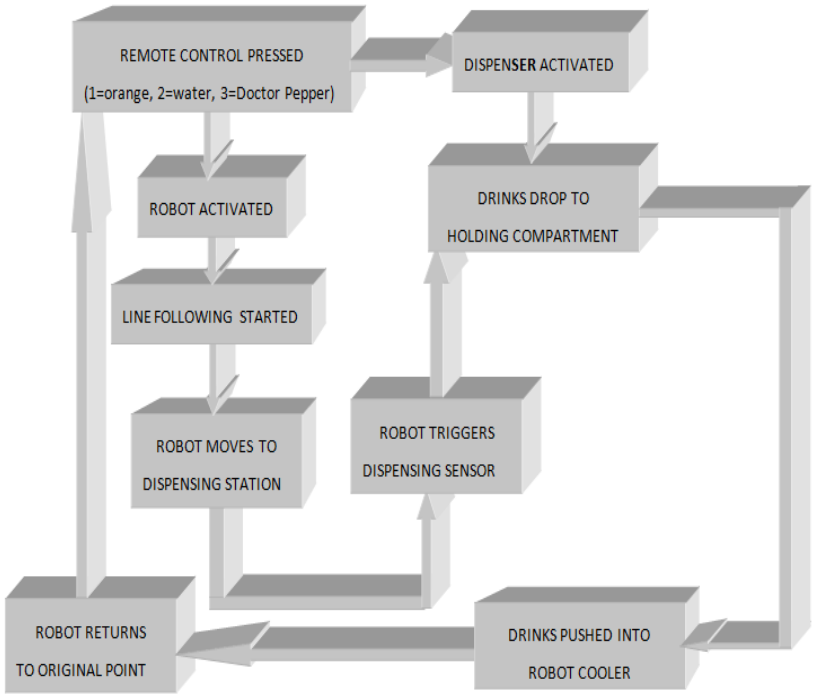

Figure 1 Block Diagram of Coolerbot Functionality

The user starts by pushing a number on the remote control. There are three choices of beverages that can be filled as pleased. For instance we can use grape soda, water and dr. pepper to correspond to keys 1, 2, and 3 respectively. The robot is activated as is the drink dispenser. The line following routine initiates and the robot follows a predefined course into the room or location of the drink dispenser. Depending on the number(s) pressed by the user, up to three choices of beverages falls into a holding compartment ready to be pushed into the robot cooler. Upon arrival at the dispensing station, the robot's presence triggers a sensor circuitry which activates the pushing mechanism and the drink(s) drops into the robot. The robot turns around and follows the course back to the starting point where the user is awaiting his/her drink.

Most of the project is constructed from components that were used in previous projects except for the actual robot platform and the holding dispensing station. The robot is based on a 3pi robot platform developed by Pololu Robots & Electronics. Additional structural modifications were needed to accomplish the project goals. The holding station was designed and manufactured in our laboratories and used inexpensive materials. The dispensing mechanism inside the holding station was also designed and manufactured locally.

The Coolerbot

The robot is constructed in the form of a cooler using the same type of materials used to fabricate typical coolers for outdoor activities. The main functions of the robot are to retrieve and deliver the beverages to the user. It is programmed to follow a specific path, through line tracing. For this project demonstration, a track is built using white poster board and black electrical tape. This track extended from one room to another adjacent room to emulate the living room and kitchen of a common household. Figure 2 shows a diagram with portion of the track used for navigation and the robot with the final cooler design illustration.

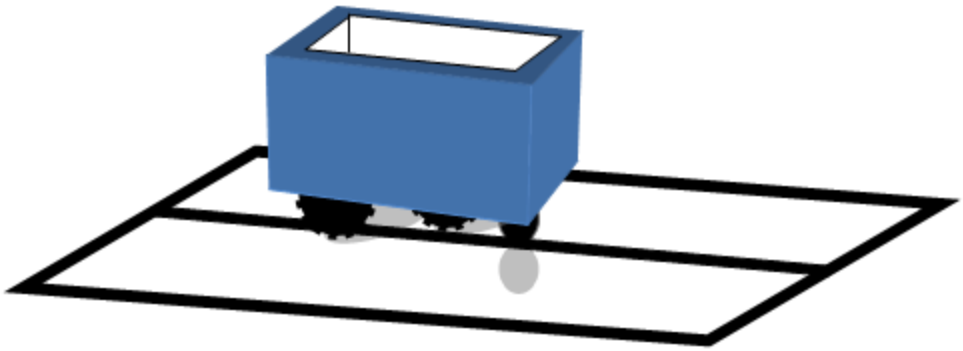

Figure 2 Section of Navigation Diagram with Coolerbot Design Illustration

3pi Platform Features and Navigation

The robot design makes use of the Pololu 3pi platform. This platform uses the Atmel ATmega168 microcontroller, running at 20 MHz. The ATmega168-based platforms feature 16 KB of flash program memory, 1 KB RAM, and 512 bytes of persistent EEPROM memory. Using the ATmegaxx8 microcontroller family makes the platform compatible with the widely used Arduino development platform [1, 7].

The 3pi requires a USB AVR Programmer which emulates an AVR ISP v2 on a virtual serial port, making it compatible with standard AVR programming software. Two additional features help with building and debugging projects: a TTL-level serial port for general-purpose communication and a SLO-scope for monitoring signals and voltage levels [1, 7].

This small, high-performance, autonomous robot platform is designed to excel in line-following. Powered by four AAA batteries and a unique power system that runs two motors at a regulated 9.25 V, the robot is capable of speeds up to 100 cm/second while making precise turns and spins that don't vary with the battery voltage. This results in highly consistent and repeatable performance of well-tuned code even as the batteries run low [1, 7, 8].

Power Management Scheme

The unique power management system employed by this design platform is depicted in Figure 3. Fluctuation on the 4 x AAA cells' voltage can be between 3.5 – 5.5 Volts. This renders it impossible to simply linearly regulate the voltage up or down to a desired 5 Volts level. The solution is to use a hybrid composed of switching and linear regulation as seen on the schematic of Figure 3.

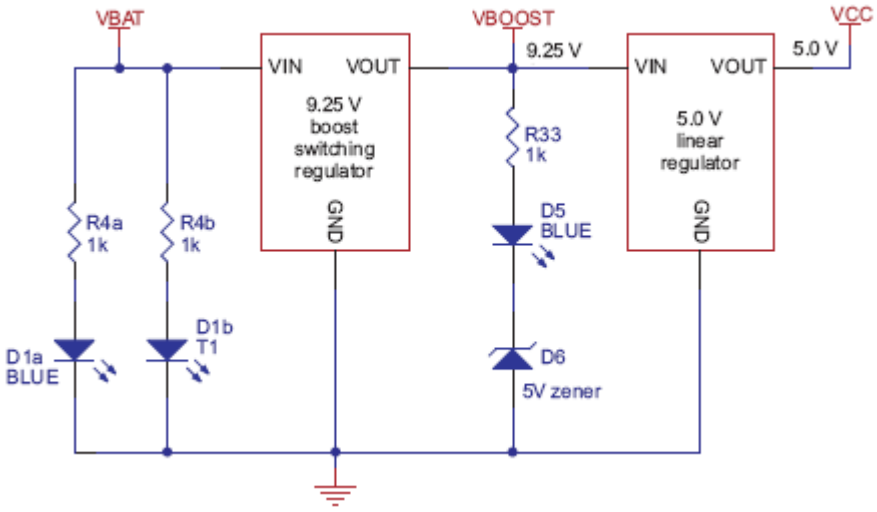

Figure 3 Power Management Scheme

A switching regulator boosts the battery voltage up to 9.25 V (Vboost), and a linear regulator regulates Vboost back down to 5 V (VCC). The voltage Vboost powers the motors and the IR LEDs used in the line sensing scheme, while VCC is used for the microcontroller functionality and digital I/O signals. By using Vboost for the motors and sensors gives the robot three unique performance advantages over typical robot platforms, which use direct battery connections to motors and sensors : (1) a higher voltage means more power for the motors, without requiring more current and/or a larger motor driver; (2) since the voltage is regulated, the motors will run at the same speed as the batteries drop from 5.5 down to 3.5 V; (3) at 9.25 V, all five of the IR LEDs can be powered in series so that they consume the lowest possible amount of power. (Note that the LEDs can be switched on and off to save even more power). In addition, a simple circuit for monitoring battery voltage is built into the 3pi. It is formed by a three resistors voltage divider circuit that outputs a voltage equal to two-thirds of the battery voltage, which will always be safely below the main microcontroller's maximum analog input voltage of 5 V [1, 7, 9].

Digital I/Os and Sensors

The microcontroller on the 3pi has a number of pins which can be configured as digital inputs or outputs. They are read by the program as a 1 or a 0 depending on whether the voltage is high (above about 2 V) or low. Figure 4 shows the circuit diagrams for the two cases where the digital I/Os are used in this project [4, 5, 7].

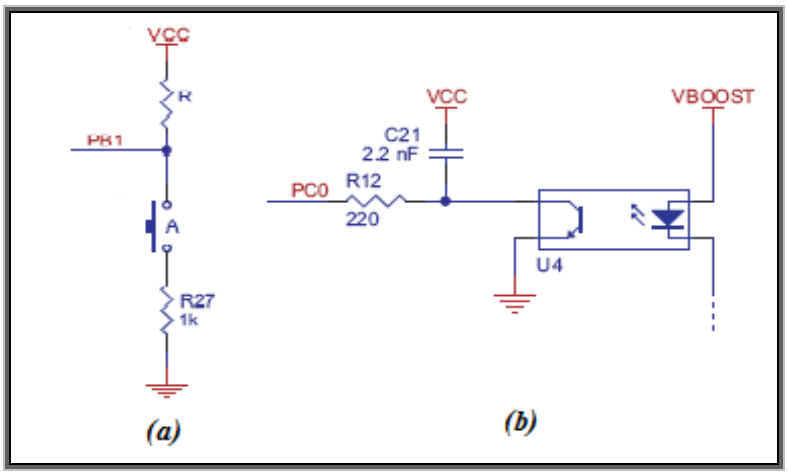

Figure 4 Digital I/Os circuit diagrams for (a) Pushbuttons and (b) Reflectance Sensor

When the pushbutton is connected to a digital I/Os it can be used as a reset or start up control signal. In Figure 4 (a) pin PB1 is connected to VCC through the pull-up resistor R (20-50 k) which sets the voltage on the input pin to 5 V, so it reads as a digital 1. Pressing the button connects the input to ground (0 Volts) through a 1 k resistor, which is much lower than the value of R. This sets the input voltage very close to 0 V, so the pin reads as a digital 0.
The digital I/Os are also used in the reflectance sensors shown in Figure 4 (b). The reflectance sensors are used for line following and deserve a more detailed analysis in order to understand the navigation code. The 3pi platform has five reflectance sensors connected from PC0 (leftmost sensor) through PC4 (rightmost sensor) digital I/O pins. The state of the pin is determined by first charging the capacitor followed by discharging it and measuring the discharging time. If the discharging time is small, it means that the phototransistor has a good connection to ground. Since the phototransistor is turned on with high reflectivity, smaller discharging time means lighter surfaces and logic 0. On the other hand, if the surface has a low reflectivity the phototransistor is off and it takes longer time for the capacitor to discharge. Since the phototransistor is turned off with lower reflectivity, larger discharging time means darker surfaces and logic 1[8, 9].

Although the sensors' digital outputs can be used as digital control signals, the 3pi platform comes with libraries that contain predefined functions that can be called in ones program. To make use of these functions in the line following navigation with black line trajectory over a white background, certain sequence of coding must be followed.

First, the sensor must be initialized in order to establish a relationship between discharging time and the reflectivity of the surface. This is handled by the function *initialize ()*. This function is called once, at the beginning of the *main ()* function. Using *robot_init*(2000) results in a sensor timeout of 2000×0.4 us = 800 us. This means that the sensor values will vary from 0 (completely white) to 2000 (completely black), where a value of 2000 indicates that the sensor's capacitor took at least 800 us to discharge. By reading the sensor value one can determine the surface time

it took to discharge and thus the type of surface under the sensor. Note that this is done even when using the digital outputs directly.

Second, the sensor must be calibrated so that their reading falls within the same range. This is accomplished by using another function, the *calibrate_line_sensors()* and turning the robot to the right and left on the line. The minimum and maximum values read during this time are stored in RAM. This allows the *read_line()* function to return values that are adjusted to range from 0 to 1000 for each sensor, even if some of the sensors respond differently than the others.

Third, since each sensor is calibrated to read a range between 0 and 1000 a convoluted technique is used to change the range from 0 to 4000 so that the five sensors ranges can be used continuously to determine the location of the robot with respect to the centered line starting with all sensors placed to the right of the line which corresponds to the lower end 0. As each sensor cross the line from left to right, the reading increases by 1000. In general, if you are using *N* sensors, a returned value of 0 means the line is on or to the outside of sensor 1, and a returned value of 1000 * (*N*-1) means the line is on or to the outside of sensor *N*-1. The value returned by *read_line ()* is divided into three possible cases: (1) **0–1000:** the robot is far to the right of the line. In this case, to turn sharply left, we set the right motor speed to 100 and the left motor speed to 0; (2) **1000–3000:** the robot is approximately centered on the line. In this case, we set both motors to speed 100, to drive straight ahead; (3) **3000–4000:** the robot is far to the left of the line. In this case, we turn sharply to the right by setting the right motor speed to 0 and the left motor speed to 100[1, 7, 8, 9, 10, 11].

The Cooler Station

The second part of the project is the beverage holder or cooler station. It is designed to resemble a soda machine. Two views of the design are depicted in Figure 5: (a) Front View and (b) side view.

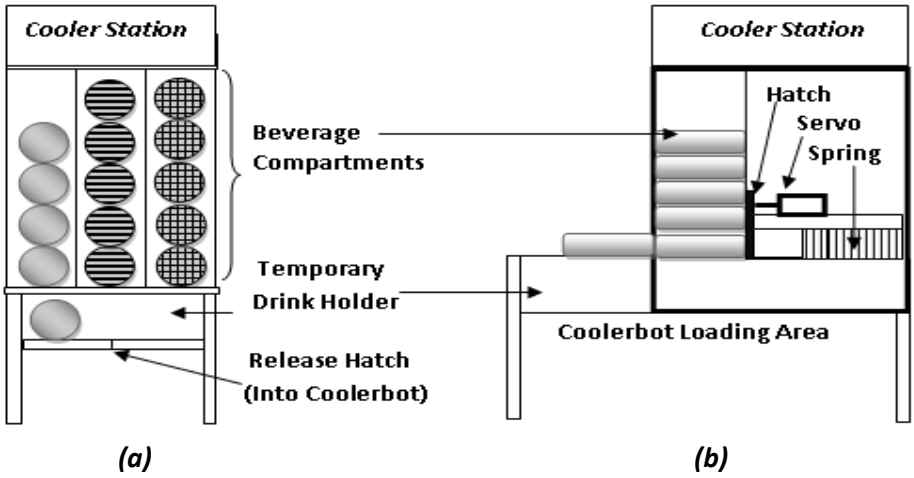

Figure 5 Cooler Station Showing (a) Front View and (b) Side View

Figure 5(a) depicts the front view of the station. Three different beverages, represented by the different textures, can be stocked in the holding compartment at any time. The beverages are simply stacked on top of each other. The design utilizes gravitational force to move the next beverage down once the bottom beverage has been released. The system is about three feet high allowing stocking to be done with ease from the top. After a beverage has been selected using the right digit on the remote control, it falls into a holding area awaiting the robot arrival to activate the release hath. Figure 5(b) is a side view of the system that can be used to explain the beverage releasing mechanism. Behind each beverage column there is a release system composed of a servo, a spring, hatch, and receiving circuitry which is implemented using a Basic Stamp board explained in more details in the control section of this paper. The hatch is attached to the servo's shaft and a spring mechanism is strategically placed behind the hatch to push beverages out once the hatch moves out of its way. Once a drink is moved out to the holding area, another one drops and is ready to be pushed. This mechanism is the same for all three drink holder columns. Another servo placed under the system will open the release hatch when the coolerbot arrives at the loading station. There is an optical sensor strategically placed to be activated by the coolerbot and send a signal to the Basic Stamp which in turns sends a signal to the servo to open the hatch to allow the drinks to be loaded and returned to the user. These mechanisms along with the initialization of the robot are controlled using sensors, transmitters, receivers and the basic stamp BS2 and constitute the control systems of this project.

The Controls

The controls system in this design includes the basic stamp BS2 microcontroller and board, infrared sensor circuits and remote control, and the programming portion for the coolerbot and basic stamp.

Basic Stamp and Infrared Circuit

The BASIC Stamp is a microcontroller with a small, specialized BASIC interpreter (PBASIC) built into ROM. It is made by Parallax, Inc. and has been popular with electronics hobbyists since the early 1990s due to its low threshold of learning and ease of use (due to its simple BASIC language). Although the BASIC Stamp has the form of a DIP chip, it is in fact a small Printed Circuit Board that contains the essential elements of a microprocessor system:

- A Microcontroller containing the CPU, a built in ROM containing the BASIC interpreter, and various peripherals
- Memory (an i²C EEPROM)
- A clock, usually in the form of a ceramic resonator
- A power supply
- External input and output

The end result is that a hobbyist can connect a 9 V battery to a BASIC Stamp and have a complete system. A connection to a personal computer allows the programmer to download software to the BASIC Stamp, which is stored in the onboard memory device. This memory is nonvolatile: it remains programmed until it is erased or reprogrammed, even after power is disconnected. The BASIC Stamp is programmed in a variant of the BASIC language, called PBASIC. PBASIC incorporates common microcontroller functions, including PWM, serial communications, I²C and 1-Wire communications, communications with common LCD driver circuits, hobby servo pulse trains, pseudo-sine wave frequencies, and the ability to time an RC circuit which may be used to detect an analog value. Once the program has been written, it is tokenized and sent to the chip through a serial cable [5, 10, 11].

The basic stamp BS2 comes integrated with a breadboard that allows the user to build circuits and access the I/O pins easily without the need to run long wire connections. The BS2 is used in this project to drive the servos and to process signals from the IR circuitries. The servos are used to control the hatches that dispense drinks into the holding area, and into the coolerbot. There are two IR circuits, one to receive the signal from the remote control and one to detect the coolerbot arrival at the cooler station. Figure 6 shows the IR circuit diagrams for the remote control signal receiving (left), the IR circuit built on the breadboard (middle), and the IR presence circuit (right) [5, 10, 11].

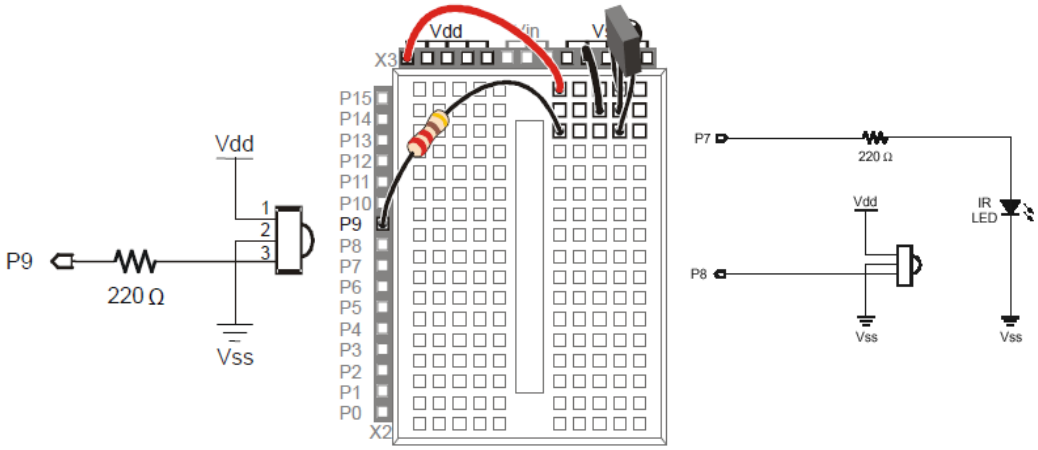

Figure 6 IR Circuits: Remote Control IR Receiver and IR Presence Sensor

The IR receiver the cooler station uses to receive the information when a key is pressed on the remote control is the same detector found in many TVs and VCRs. This detector sends a low signal whenever it detects IR flashing on/off at 38.5 kHz and a high signal the rest of the time. When the IR detector sends low signals, the processor inside a TV or VCR measures how long each of the low signals lasts. Then, it uses these measurements to figure out which key was pressed on the remote. For this project we use a SONY TV control. Most universal remotes can be configured to send messages to a SONY TV. Note that the remote control IR receiver circuit and the IR presence circuit are the same except that for the remote receiver we have a transmitter inside the remote and for the presence circuit we must build our own transmitter.

When a button on the remote is pressed, it flashes its IR LED on/off at 38.5 kHz to broadcast a code for that button. The code is produced by controlling the brief amounts of time the IR LED flashes on and off. Figure 7 shows a timing diagram example signal a SONY TV IR detector may receives when a control message from the IR remote is sent. This message consists of thirteen negative pulses that the BASIC Stamp can easily measure. The first pulse is the start pulse, which lasts for 2.4 milliseconds. The next twelve pulses will either last for 1.2 ms (binary-1) or 0.6 ms (binary-0). The first seven data pulses contain the IR message that indicates which key is pressed. The last five pulses contain a binary value that specifies whether the message is intended being sent to a TV, VCR, CD, DVD player, etc. The pulses are transmitted in least significant first order (LSB-first order), meaning the first data pulse is bit-0, the next data pulse is bit-1, and so on. If you press and hold a key on the remote, the same message will be sent over and over again with a 20 to 30 ms rest between messages [2, 3, 5].

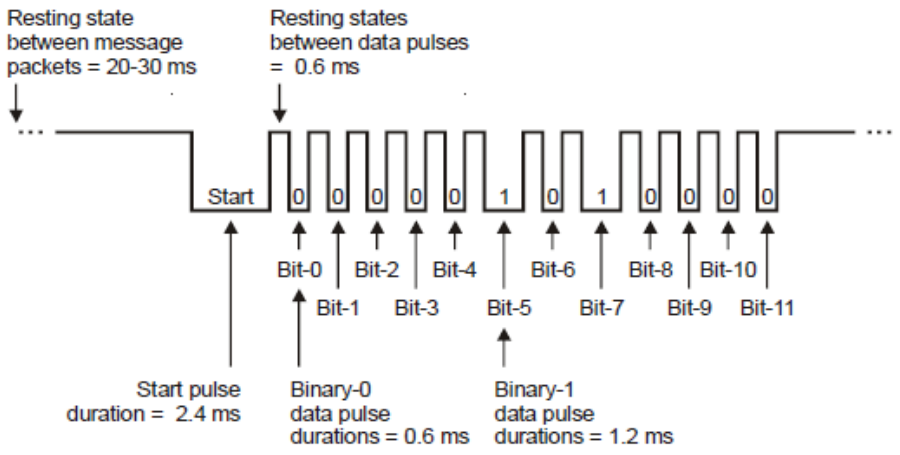

Figure 7 IR Message Timing Diagram

For this project, we must recognize when someone is pressing and holding the 1, 2, or 3 keys, and take different actions, releasing of the three different types of drinks. In order to know what signal is generated for each pressed key, we first had to characterize the IR messages sent from the remote to the receiver since our system is not exactly a SONY TV. This focuses on measuring the pulses the IR detector sends when it detects messages from the handheld remote. The IR detector sends a low signal while it detects infrared flashing on/off at 38.5 kHz. While the IR detector does not detect 38.5 kHz infrared, it sends a high signal. Before the message can be interpreted by the BASIC Stamp module, the durations of these low signals have to be measured and stored. Once the messages were characterized, a program is used to measure and capture each pulse in the pulse width modulation message sent by the infrared remote. There are a total of twelve data pulses in a given message, and each can be captured and stored [2, 3, and 5]. By programming the BASIC Stamp to capture and store these pulses, we were able to use them

in decision making. Twelve pulse measurements have to be stored separately because they are each a unique part of the message. Even so, they are related to each other. They are all pulse measurement values, and they are all going to be examined to determine which key on the remote is being pressed. The best way to store a group of related values is by using a special kind of variable called an array variable. An array variable is a group of variables that all have the same name. Declaring an array variable is similar to declaring most other variables. The first seven time array elements contain all the pulse measurements that you'll need to identify the remote's buttons. Once the elements were recorded we could then assign the pulse measurements to a specific button on the remote.

Programming

The project required programming two microcontrollers, the basic stamp BS2 and the coolerbot line following. Figure 8 shows a sample code used by the BS2 to decide which drinks to dispense. Once the proper key is pressed a signal is send from the IR circuit to BS2 microcontroller which will in turns send a corresponding signal to the proper servo to release the drink. Figure 9 shows a sample code used for line following. The commands and program sequences are explained in the sensors section.

```
DO ' Beginning of main loop.
DO ' Wait for rest between messages.
PULSIN 9, 1, time(0)
LOOP UNTIL time(0) > 1000
PULSIN 9, 0, time(0) ' Measure/store data pulses.
PULSIN 9, 0, time(1)
' Decide which maneuver to execute depending on the combination
' of pulse durations stored in the first two pulse measurements.

' Key 1
' This controls the Mechanism for Grape Soda
IF (time(1) < 500) AND (time(0) < 500) THEN
FOR counter = 1 TO 250                          ' Number of pulses - run time.
PULSOUT 12, 650
NEXT
FOR counter = 1 TO 50
PULSOUT 12, 850
NEXT
RETURN                                          'Return to the main program


' Key 2
' This controls the Mechanism for Dr. Pepper
ELSEIF (time(1) < 500) AND (time(0) > 500) THEN
FOR counter = 1 TO 250                          ' Number of pulses - run time.
PULSOUT 13, 850
NEXT
FOR counter = 1 TO 50
PULSOUT 13, 650
NEXT
RETURN                                          'Return to the main program
```

Figure 8 Basic Stamp BS2 Sample Code for the IR Remote Control

```
// Always wait for the button to be released so that robot doesn't
// start moving until your hand is away from it.
wait_for_button_release(BUTTON_B);
delay_ms(1000);

// Auto-calibration: turn right and left while calibrating the
// sensors.
for(counter=0;counter<80;counter++)
{
    if(counter < 20 || counter >= 60)
        set_motors(40,-40);
    else
        set_motors(-40,40);

    // This function records a set of sensor readings and keeps
    // track of the minimum and maximum values encountered.  The
    // IR_EMITTERS_ON argument means that the IR LEDs will be
    // turned on during the reading.
    calibrate_line_sensors(IR_EMITTERS_ON);

    // Since the counter runs to 80, the total delay will be
    // 80*20 = 1600 ms.
    delay_ms(20);
}
set_motors(0,0);

// Display calibrated values as a bar graph.
while(!button_is_pressed(BUTTON_B))
{
    // Read the sensor values and get the position measurement.
    unsigned int position = read_line(sensors,IR_EMITTERS_ON);

    // Display the position measurement, which will go from 0
    // (when the leftmost sensor is over the line) to 4000 (when
    // the rightmost sensor is over the line) on the robot, along
    // with a bar graph of the sensor readings.  This allows you
    // to make sure the robot is ready to go.
    clear();
    print_long(position);
    lcd_goto_xy(0,1);
    display_readings(sensors);

    delay_ms(100);
}

// This is the main function, where the code starts.
int main()
{
    unsigned int sensors[5]; // an array to hold sensor values

    // set up the robot
    initialize();

    // This is the "main loop" - it will run forever.
    while(1)
    {
        // Get the position of the line.  Note that we *must* provide
        // the "sensors" argument to read_line() here, even though we
        // are not interested in the individual sensor readings.
        unsigned int position = read_line(sensors,IR_EMITTERS_ON);

        if(position < 1000)
        {
            // We are far to the right of the line: turn left.

            // Set the right motor to 100 and the left motor to zero,
            // to do a sharp turn to the left.  Note that the maximum
            // value of either motor speed is 255, so we are driving
            // it at just about 40% of the max.
            set_motors(0,100);

            // Just for fun, indicate the direction we are turning on
            // the LEDs.
            left_led(1);
            right_led(0);
        }
        else if(position < 3000)
        {
            // We are somewhat close to being centered on the line:
            // drive straight.
            set_motors(100,100);
            left_led(1);
            right_led(1);
        }
        else
        {
            // We are far to the left of the line: turn right.
            set_motors(100,0);
            left_led(0);
```

Figure 9 Sample Code Used by the Coolerbot for Line Following

Conclusions

A fully automated drink dispenser for the physically impaired has been designed and implemented as part of a senior capstone project. This design is a prime example of how independent platforms normally used by hobbyist can be integrated to achieve designs that are useful to the human kind. The design integrates the 3pi platform with the BS2 board, IR technology, the common TV remote control, and basic servos to arrive to an integrated system to assist those that are less fortuned by providing them with a way to fulfill their thirst. Although some of these technologies were in place, the novelty of this project was to bring them together in a creative and useful manner. This design can be modified to assist the physically impaired in other areas such as a snack machine, basic retrieval of objects and many other necessities.

References:

1. http://www.pololu.com
2. http://www.radio-electronics.com/
3. http://www.ieee.org/portal/site/tionline/menuitem.130a3558587d56e8fb2275875bac26c8/index.jsp?&pName=institute_level1_article&TheCat=1008&article=tionline/legacy/inst2005/may05/5w.fhistory.xml&.
4. http://www.ti.com/corp/docs/kilbyctr/jackbuilt.shtml
5. www.parralax.com
6. http://www.wikipedia.org
7. http://www.atmel.com
8. http://www.avrbeginners.net/
9. http://www.avr-asm-tutorial.net/
10. http://www.lancos.com/prog.html
11. http://www.kpsec.freeuk.com/
12. http://www.williamson-labs.com/home.htm