# Contents

## Current Status

1. Work Completed

## 1. Construction

## 2. Servo Control Unit

## 3. Servo Controller Protocol: SCP

## 4. Brain

## Current Work

## Future Work

## Appendices

## Time Line
Current Status

Construction is complete. The Brain and SCU boards are built and mounted to the snake. The wiring between the Brain, the SCUs, and the servos is installed. The tethered power supply line is connected. The microcontrollers are programmed and we are ready to begin testing the motion.

Compared to our original time line we are a month behind. This is due to over optimistic estimates for construction time. The entire construction time from beginning to the end of prototype 1a was greatly underestimated and has cost us valuable developmental time for further prototypes. We have thus chosen to drop the pursuit of prototype 2. At this time, such a goal would be improbable. We are instead going to focus on the completion of prototype 1a and the implementation of prototype 1b.

Work Completed

1 Construction

Construction of prototype 1a has been completed. Over the break, several improvements were made to the snake in order to make it stronger. “L” brackets were mounted to the universal joints to stabilize them. Prior to the fledgling brackets, the backbone could be rotated torsionally which debilitated the motion of the snake. Bolts were also added to all the joints to prevent them from twisting. All of the segments were aligned making their wheels straight and level with one another. The head and remaining segments were also constructed and attached to the snake.

The Brain and SCUs were implemented on pcb boards. The SCU schematic and pcb layout are shown in Figure 1. The brain schematic is shown in Figure 2.

![Figure 1: Schematic and board layout of Servo Control Unit](image)
The SCU has been fully implemented. We originally believed that the servos would be controlled using the PIC16F628’s HPWM function. A decoder would split the output signal between the four servos. We discovered that the lengthy propagation delay of the decoder scrambled the pulses. We tried to use FET transistors as switches to split the signal but yielded similar results. It was then discovered that PicBasic has a function called PULSEOUT that puts a pulse of a specified width on any output pin. We were able to control four servos simultaneously by rotating the PULSEOUT function through four output pins of the pic. Each servo was connected to a different pin.

The PULSEOUT function generates a pulse. The pulse width is measured in increments of 10 $\mu$sec. The servos accept pulses between the range of 0.9ms to 2.1ms with the center position at 1.5ms.

PicBasic also has serial output and serial input functions. These functions were used to implement the SCP. After extensive testing, communication was established between the brain and the SCU using a baud rate of 1200 and bit spacing of .1ms. With this, the first working prototype of the SCU and the brain were created and are shown in the files “first_SCU.bas” and “first_brain.bas.”

During testing, we discovered that the interrupt service routine on the SCU was too long. When an interrupt is initiated, the SCU quits sending pulses to execute the ISR. In theory,
if the routine is short enough, it can be executed without any notice. This was not the case since our lengthy routine caused the servos to jump and move roughly. In order to shortened the ISR the communication between the SCU and the Brain was shortened.\footnote{Please see Section 3 for details.}

When the SCU receives new positions for its servos, the data is converted into pulse lengths. Each position command is a byte of data representing a number between '0' and '255'. The '0' position tells the SCU to output a minimum pulse length of 0.9ms. The '255' position tells the SCU to output a maximum pulse length of 2.1ms. The SCU can not use these arbitrary positions by themselves and needs to convert them to meaningful pulse lengths. To perform this duty, the following equation is used:

\[
pulse = \frac{\text{position}}{3} + 108
\]

This equation provides a pulse between 108ms and 193ms with the center at exactly 150ms. The range of the pulses is shortened because the servos bind when they are near their maximum and minimum positions.

3 Servo Controller Protocol: SCP

The SCP standardizes the communication between the Brain and the SCUs. Firstly, a numbering scheme (Figure 3) is organized that identifies each SCU and servo. The six SCUs control the 24 servos of the snake. The SCUs are labeled with numbers from 1 to 6. SCU '1' is located closest to the head. SCU '6' is located closest to the tail. The four servos attached to each SCU are labeled 1 through 4. Servo '1' is nearest to the head while servo '4' is nearest to the tail.

The SCP specifies serial communication of position commands to the SCU at a baud rate of 1200. This data is passed on a dedicated Serial Data Bus.\footnote{Labeled SCU.TDX bus in Figures 1 and 2.} The Brain initiates serial communication by sending an interrupt signal to the SCUs on a dedicated Interrupt bus.\footnote{Labeled SCU.INT bus in Figures 1 and 2.} This pulse triggers all the SCUs to enter the Interrupt Service Routine(ISR). Once they enter the routine, they begin listening to the Serial Data Bus for commands from the Brain. The following explanation is shown in a flowchart in Figure 6 in the appendix.

The Brain sends an initial dummy byte, which provides a delay to allow the SCUs time to enter the ISR. As shown in Figure 4, the Brain then sends the character “F” followed
by the first byte of information. The character gives the SCU a reference point to start storing information. The first byte of data contains the controller ID of the SCU to update. If the controller id of the SCU does not match the transmitted controller id, the SCU quits the ISR and continues with the main program. The SCU with the right id continues listening for new servo positions. The Brain sends the character “F” followed by the four new positions to the SCU (Figure 5). The SCU receives these positions, stores them, and terminates communication with the Brain. The Brain can start communication again any time by triggering an interrupt. These interrupts and data exchange processes can repeat indefinitely.

The SCP defines the position command in one byte. The minimum position is indicated with a command of 0 while the maximum position is indicated with a command of 255. The command of 128 indicates the middle position. The range of positions are from 0 to 255 by increments of 1. Given this command, the SCU must translate the position into a pulse width according to the specification of the servos.

4 Brain

The first Brain is now complete. It provides basic motion control and can communicate with the SCUs. Motion control is achieved using a serpenoid function.

\[
\phi = \alpha \sin \left( \omega t + (i - 1)\beta \right) + \gamma
\]  

(2)

This function is designed for floating point numbers. Since the microcontroller used in the brain can only perform integer arithmetic, the serpenoid function must be approximated. The parameters are separated into an integer part and decimal parts. For example \( \alpha \) is separated into an integer part \( a \), decimal part \( a_{10} \), decimal part \( a_{100} \), etc. \( a \) is a whole number. \( a_{10} \) is the 10ths place digit and is divided by 10. \( a_{100} \) is divided by 100 and so forth. Hence

\[
\phi = a \sin \left( \ldots \right) + \frac{a_{10}}{10} \sin \left( \ldots \right) + \frac{a_{100}}{100} \sin \left( \ldots \right) + \ldots + \gamma
\]  

(3)

Accordingly, each parameter (\( \alpha, \beta, \gamma, \) and \( \omega \)) can be expanded to achieve a desired level of precision.

The Brain has a main routine that calls two other subroutines: “trace” and “updateSCUs”. The main routine repeatedly calls these subroutines. The “trace” subroutine facilitates the motion control. The new positions of the servos are calculated from the serpenoid function. The function is evaluated using the PicBasic SIN function, which returns a two’s
complement result of sine. The domain of the function is from 0 to 255. Essentially, $2\pi$ or 360 degrees is mapped to 255. This is nice, since the function is periodic over 1 byte, which is a simple representation to communicate with the SCUs.

It is very important to keep in mind that the result is in two’s complement, because the Servo Controller Protocol specifies the minimum position as 0 and the maximum position as 255. The middle position is 128. But, in two’s complement the minimum and maximum positions are -127 and 127 respectively. The two’s complement middle position is at 0. The two’s complement representation can be transformed in the Protocol’s position command by simply adding 127 to the result.

The “trace” subroutine defines horizontal motion, where adjacent servos work in opposite directions. The joints of the snake are formed by two servos. The joints bend from side to side when the two servos work in opposite directions. According to the Servo Controller Protocol, one of the two adjacent servos is labeled with an even number, and the other servo is labeled with an odd number. For example consider the first two servos. They are labeled ‘1’ and ‘2’. The ”trace” subroutine calculates the desired position of these servos. For horizontal motion, positions of servos ‘1’ and ‘2’ have the following relationship:

$$\text{position}[2] = 255 - \text{position}[1]$$

(4)

If “position[1]” moves toward the max position, “position[2]” moves the same amount but toward the min position. Consider the trivial case where “position[1]” is in the minimum position, which is the command 0. Then,

$$\text{position}[2] = 255 - 0 = 255$$

(5)

Thus, “position[2]” is in the maximum position, which is the command 255. After the new positions are calculated, the main routine sends these commands to the SCUs by calling “updateSCUs” subroutine.

**Current Work**

We are currently working on the brain of prototype 1a. Our goal is to achieve simple horizontal motion with no sensory intelligence. We are focused on getting the serpenoid function to output correct servo positions that will propel the snake forward. As described above, working with only integer arithmetic is challenging. We are also starting to design the peripheral modules and how we plan to implement them into prototype 1b. This includes designing the Peripheral Bus Protocol(PBP).

**Future Work**

As soon as successful movement from prototype 1a is achieved, we will start testing the overall design of prototype 1a. We will find out how all the modules of our design work together. Any improvements will be made at this time. We will then start the development of prototype 1b. In this prototype we will enhance the intelligence of the snake by adding
feedback from its environment through the use of peripheral modules. Our first two modules to be added will be the infrared collision detector and a velocity meter. The infrared collision detector will inform the snake if it is about to collide with an obstacle and the velocity meter will inform the snake how fast it is moving and what direction it is moving in.
Appendix

Flow Chart of SCU

Figure 6: New Flow Chart of SCU
### Time Line

<table>
<thead>
<tr>
<th>ID</th>
<th>Task Name</th>
<th>Start</th>
<th>Finish</th>
<th>Duration</th>
<th>% Complete</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Prototype 1a</td>
<td>1/16/2004</td>
<td>3/12/2004</td>
<td>32d</td>
<td>95%</td>
</tr>
<tr>
<td>2</td>
<td>Implementation</td>
<td>1/16/2004</td>
<td>2/25/2004</td>
<td>25d</td>
<td>99%</td>
</tr>
<tr>
<td>3</td>
<td>Construction</td>
<td>1/16/2004</td>
<td>2/13/2004</td>
<td>21d</td>
<td>163%</td>
</tr>
<tr>
<td>4</td>
<td>Electrical</td>
<td>1/16/2004</td>
<td>2/25/2004</td>
<td>29d</td>
<td>90%</td>
</tr>
<tr>
<td>5</td>
<td>Testing</td>
<td>1/16/2004</td>
<td>2/13/2004</td>
<td>21d</td>
<td>163%</td>
</tr>
<tr>
<td>6</td>
<td>Analysis</td>
<td>2/25/2004</td>
<td>3/12/2004</td>
<td>4d</td>
<td>6%</td>
</tr>
<tr>
<td>7</td>
<td>General Faults</td>
<td>2/25/2004</td>
<td>3/12/2004</td>
<td>4d</td>
<td>6%</td>
</tr>
<tr>
<td>9</td>
<td>Prototype 1b</td>
<td>3/1/2004</td>
<td>4/30/2004</td>
<td>45d</td>
<td>6%</td>
</tr>
<tr>
<td>11</td>
<td>Mechanical</td>
<td>2/20/2004</td>
<td>3/10/2004</td>
<td>14d</td>
<td>6%</td>
</tr>
<tr>
<td>12</td>
<td>Electronics</td>
<td>2/20/2004</td>
<td>3/10/2004</td>
<td>14d</td>
<td>6%</td>
</tr>
<tr>
<td>13</td>
<td>Intelligence</td>
<td>2/20/2004</td>
<td>3/10/2004</td>
<td>14d</td>
<td>6%</td>
</tr>
<tr>
<td>17</td>
<td>Analysis</td>
<td>4/1/2004</td>
<td>4/20/2004</td>
<td>14d</td>
<td>6%</td>
</tr>
<tr>
<td>18</td>
<td>General Faults</td>
<td>4/1/2004</td>
<td>4/20/2004</td>
<td>14d</td>
<td>6%</td>
</tr>
<tr>
<td>19</td>
<td>Improvements</td>
<td>4/1/2004</td>
<td>4/20/2004</td>
<td>14d</td>
<td>6%</td>
</tr>
<tr>
<td>20</td>
<td>Final Analysis</td>
<td>4/20/2004</td>
<td>4/28/2004</td>
<td>14d</td>
<td>6%</td>
</tr>
</tbody>
</table>