State Machine Diagram

Design Solution

Ultrasonic Sensors

Design Problem Specifications

Navigating Corners

Robotic Wall-Following Edge Bots Using Ultrasonic Sensors

Design Problem

Executing a Turn

Adjustment States

SCOBY Alterations

7-Segment Display

LCD Display

LED Arrays

Assignments

Problems Encountered

Results

Discussion

Strengths

- Completes course
- Avoids collisions
- Speed control
- Informative display

Weaknesses

- Slow
- Inconsistent and inefficient algorithms

Proposed vs. Actual Design

Future Development

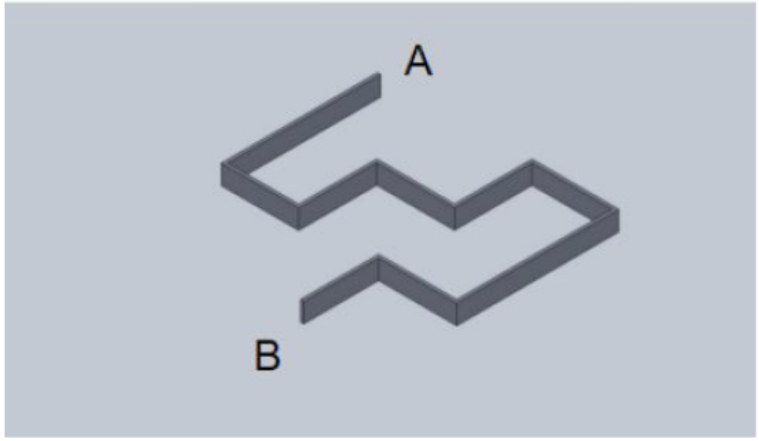
Robotic Wall Following Algorithm Using Ultrasonic Sensors

Christopher Hood
Stuart Kent
Matthew McGraw
Anya Skomorokhova

ECE 2031 Digital Design Laboratory L06
Georgia Institute of Technology

December 7, 2011

Design Problem



Objective: Design a SCOMP wall-following program that will enable the Amigobot to follow walls with inner and outer corners at a distance of 20 cm from the wall

Design Problem Specifications

1. Use an eight bit velocity command

2. Control position by reading the cumulative rotation counter of the wheel

3. Use velocity feedback and position feedback from the wheels via the existing optical encoder peripheral

4. Use existing sonar and velocity control peripherals to issue commands to each wheel

Design Solution

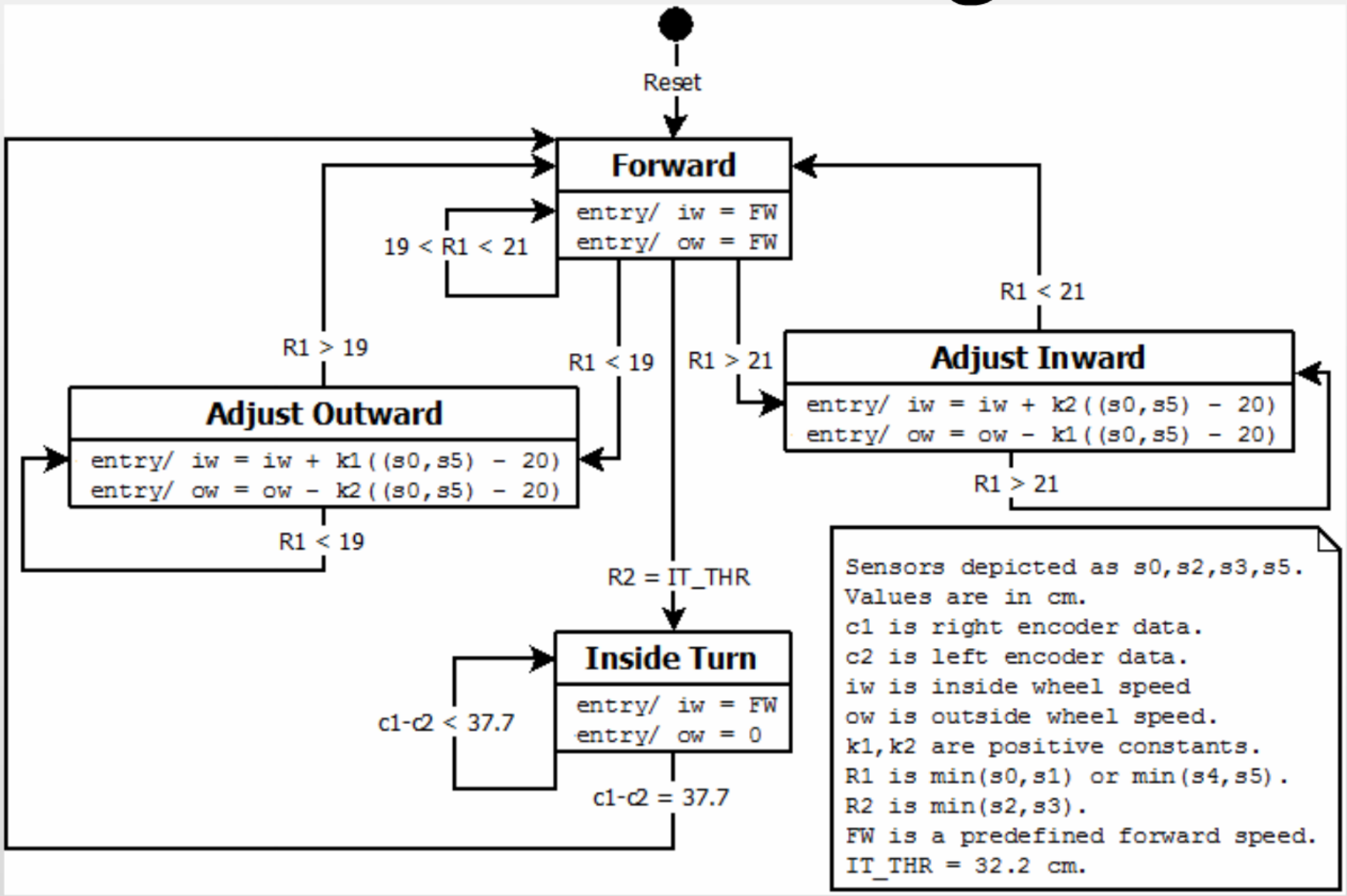
State Machine Design

- Four states: Forward, Adjust Inward, Inside Turn, Adjust Outward
- Inputs are sensor readings
- Outputs are direction instructions
- Switch is used to toggle between following left and right walls

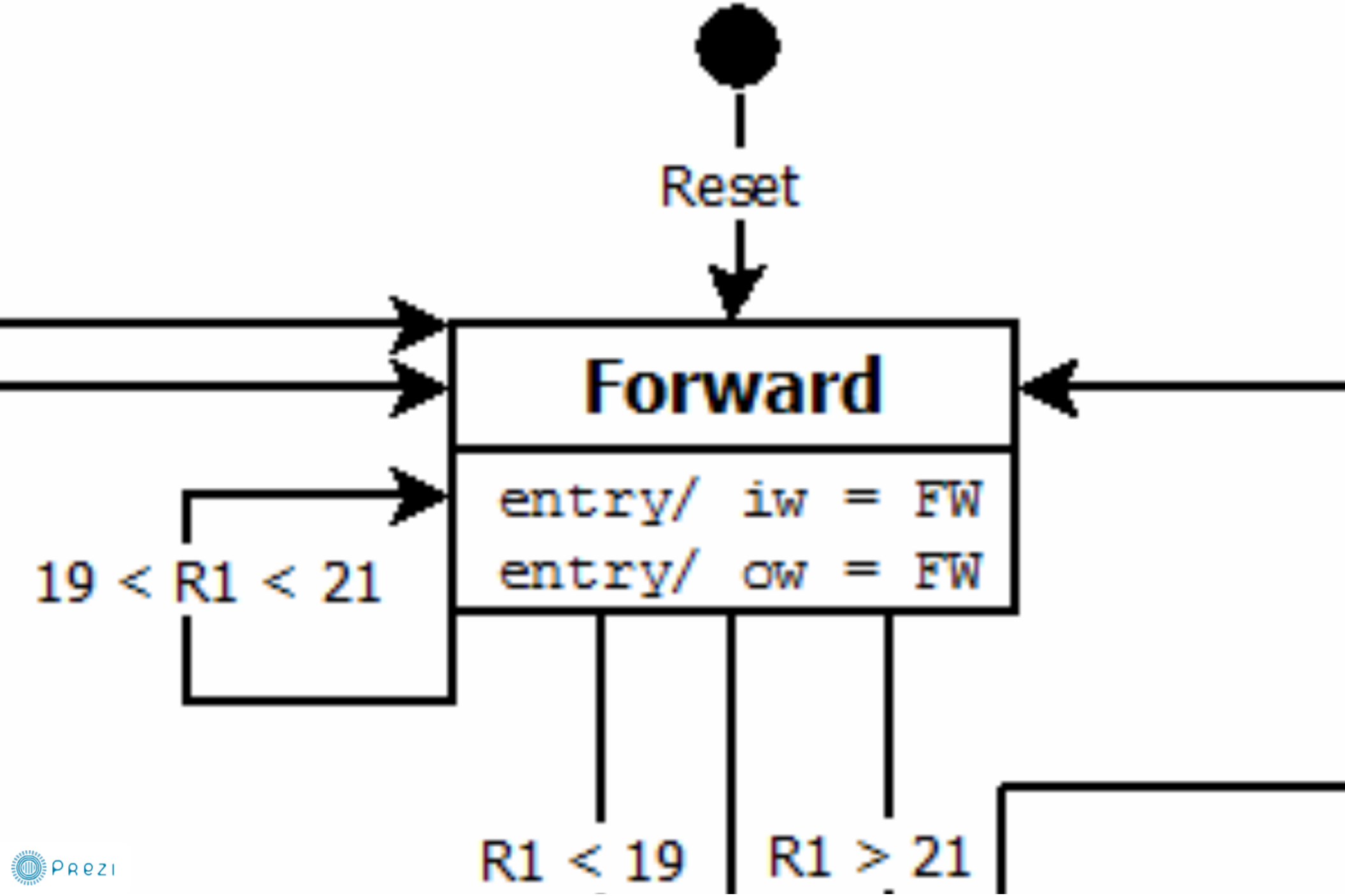
Display additions

- LCD displays status of program
- 7-segment displays show velocities of wheels

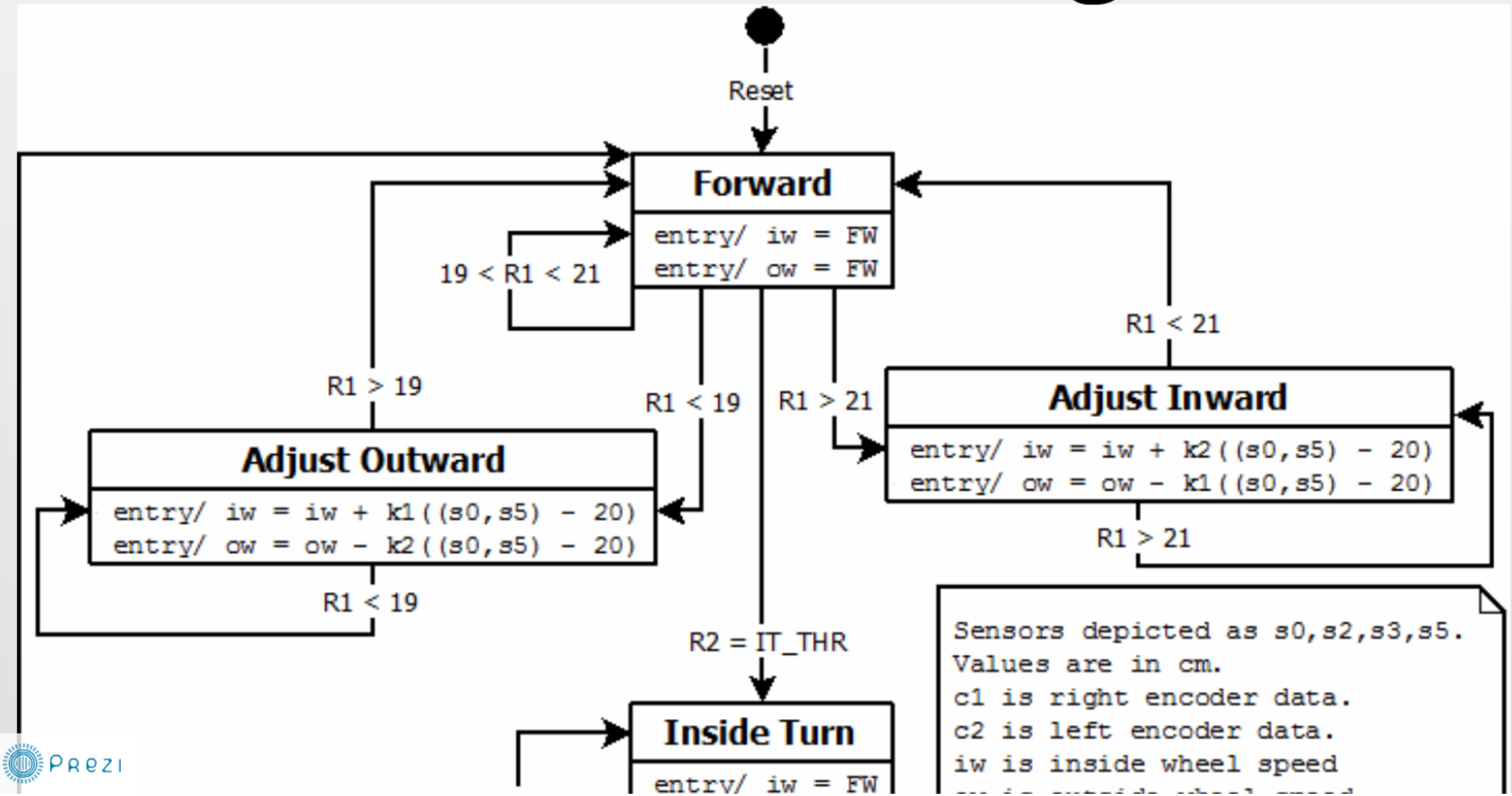
State Machine Diagram

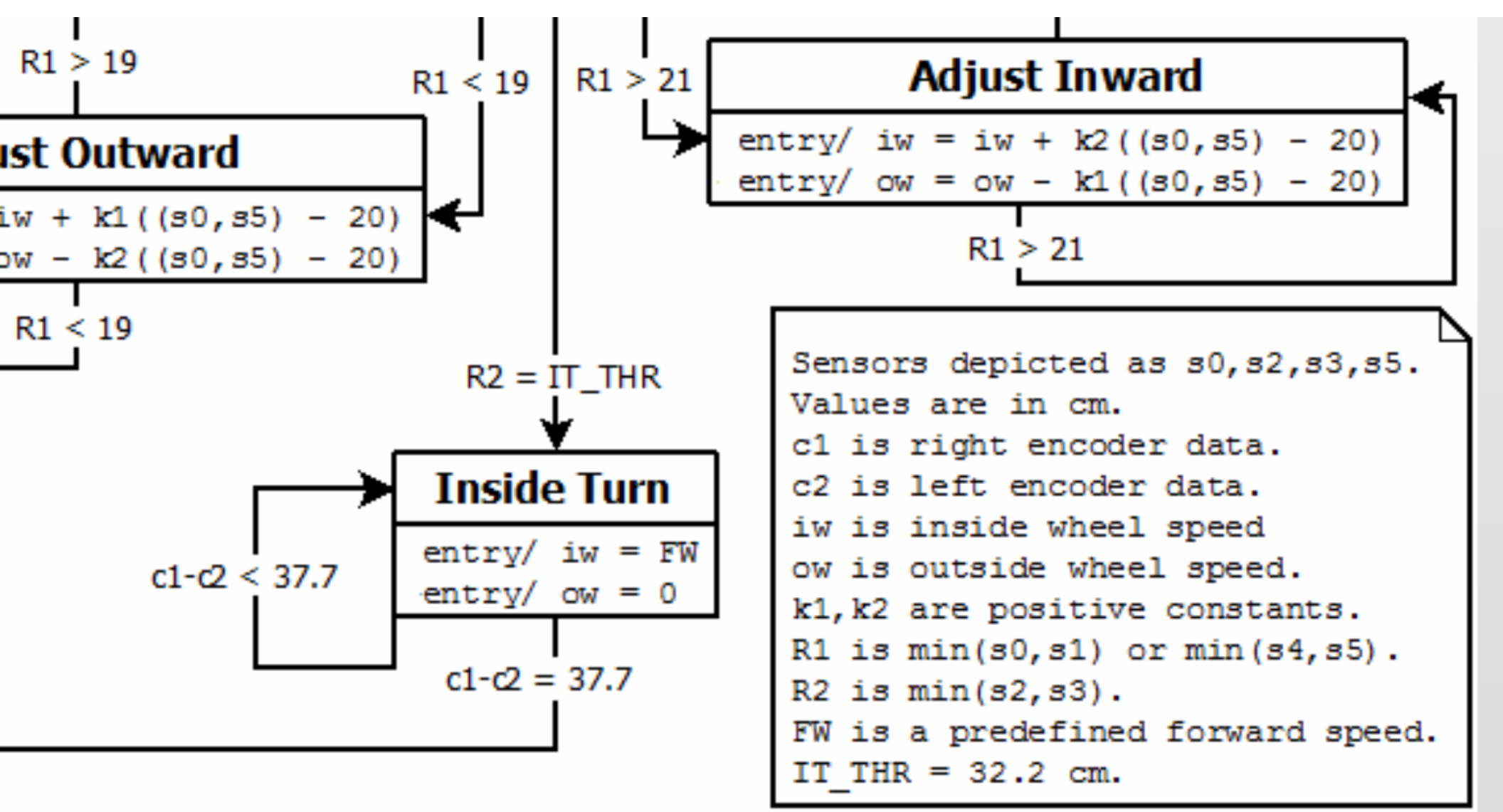


Machine Dia



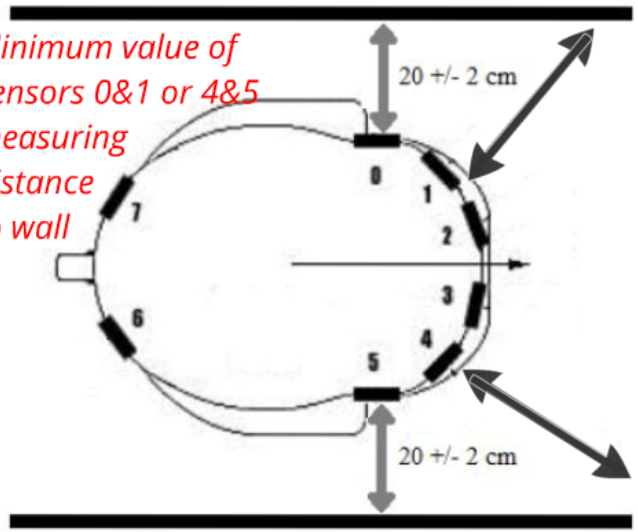
State Machine Diagram



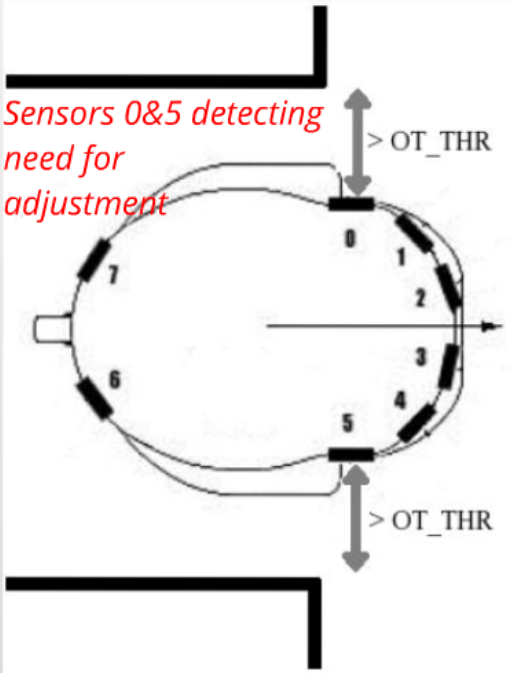

R1 > 19
R1 < 19
R1 > 21
ust Outward
iw + k1((s0,s5) - 20)
ow - k2((s0,s5) - 20)
R1 < 19
Adjust Inward
entry/ iw = iw + k2((s0,s5) - 20)
entry/ ow = ow - k1((s0,s5) - 20)
R1 > 21
R2 = IT_THR
Inside Turn
entry/ iw = FW
entry/ ow = 0
c1-c2 < 37.7
c1-c2 = 37.7
Sensors depicted as s0,s2,s3,s5.
Values are in cm.
c1 is right encoder data.
c2 is left encoder data.
iw is inside wheel speed
ow is outside wheel speed.
k1,k2 are positive constants.
R1 is min(s0,s1) or min(s4,s5).
R2 is min(s2,s3).
FW is a predefined forward speed.
IT_THR = 32.2 cm.

Ultrasonic Sensors

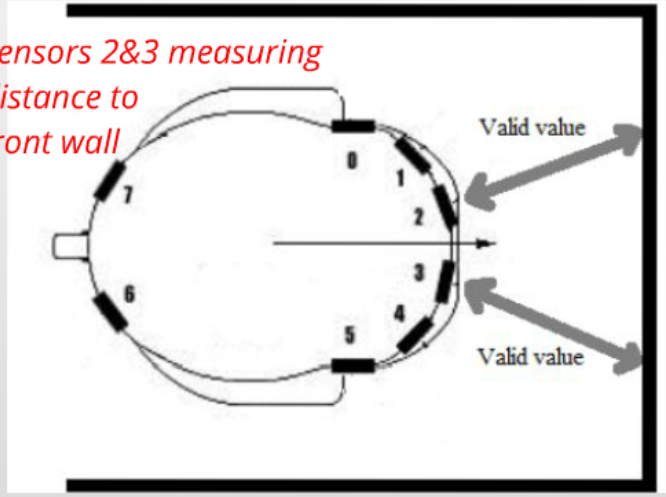
Minimum value of Sensors 0&1 or 4&5 measuring distance to wall



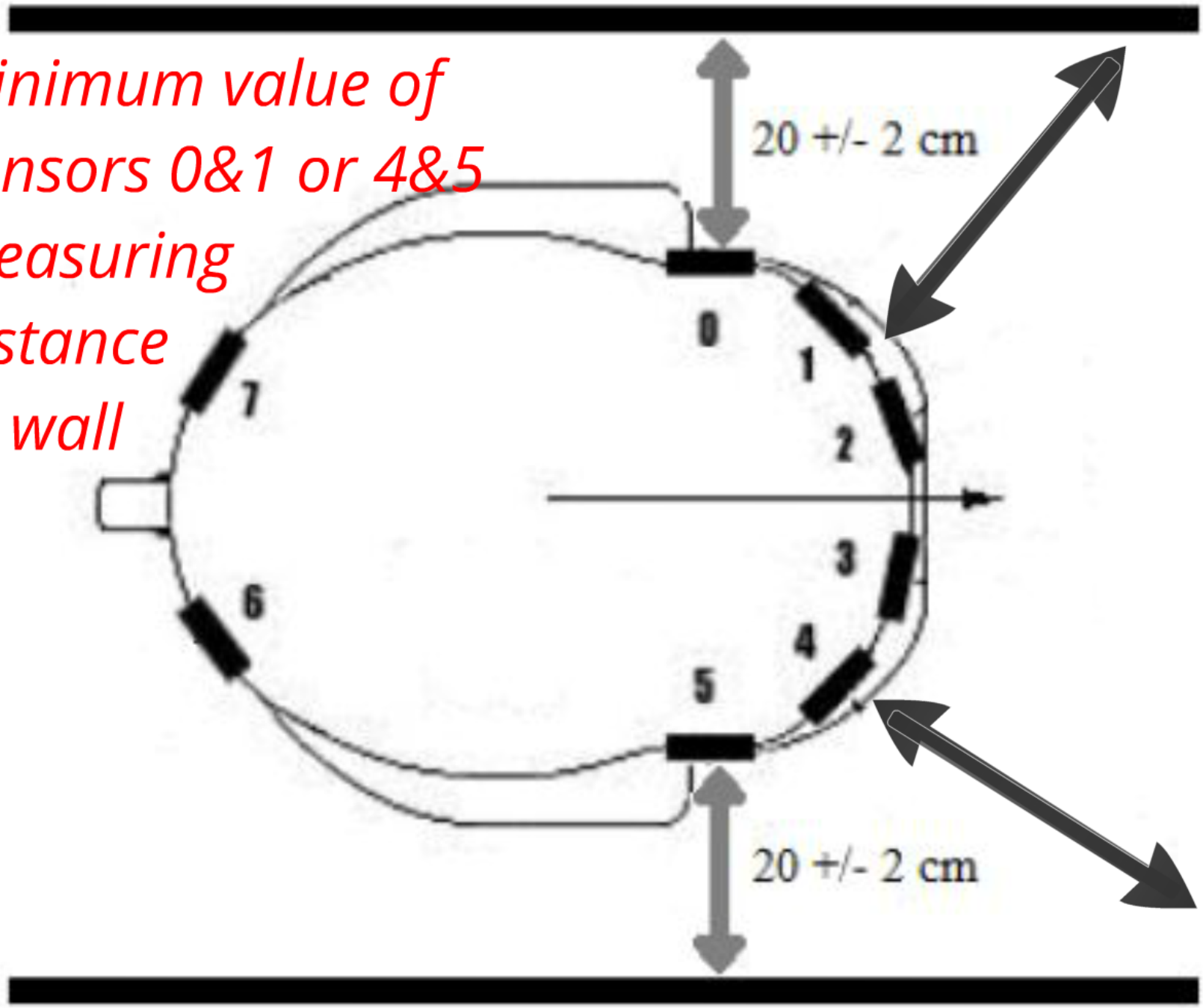
Sensors 0&5 detecting need for adjustment

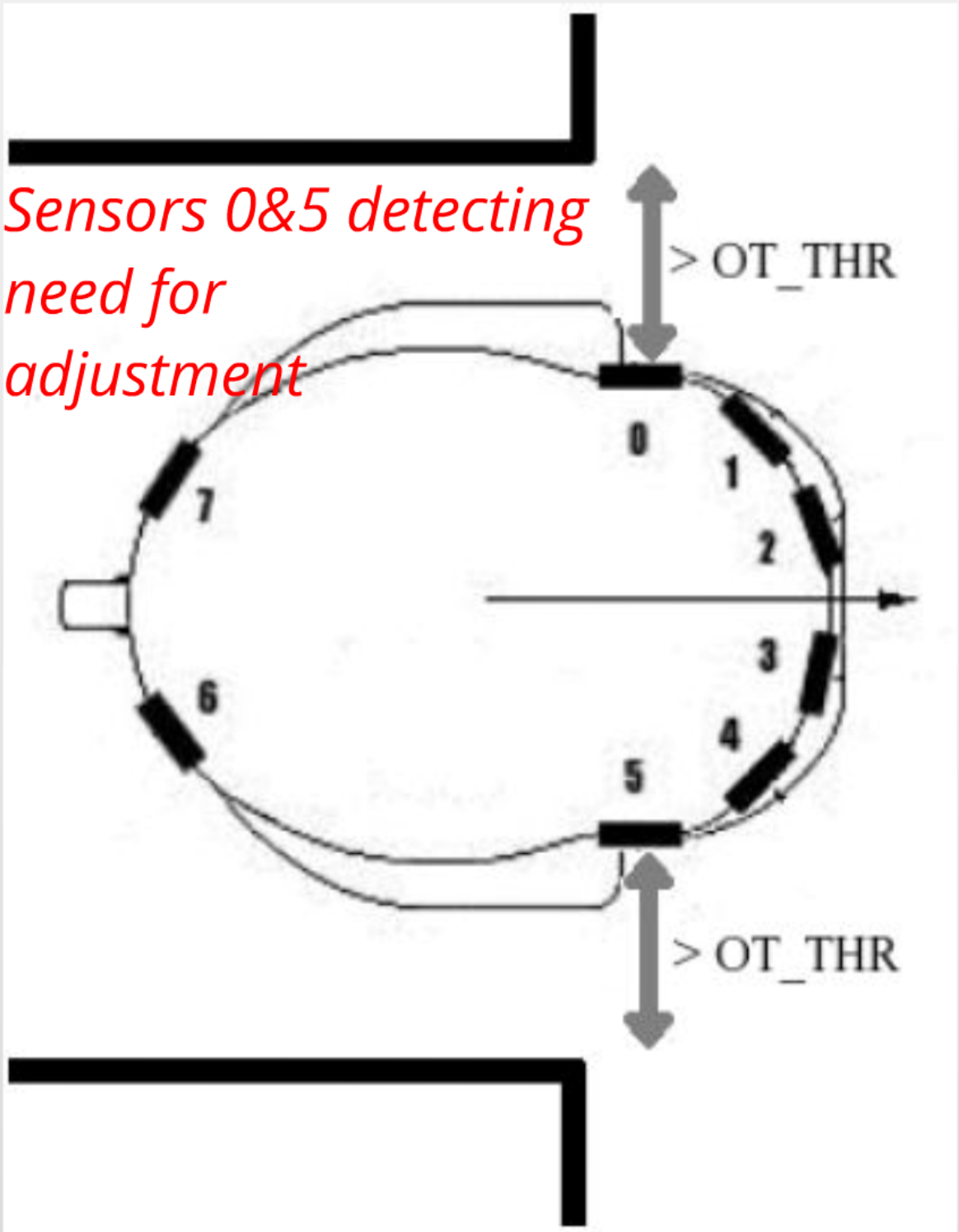


Sensors 2&3 measuring distance to front wall

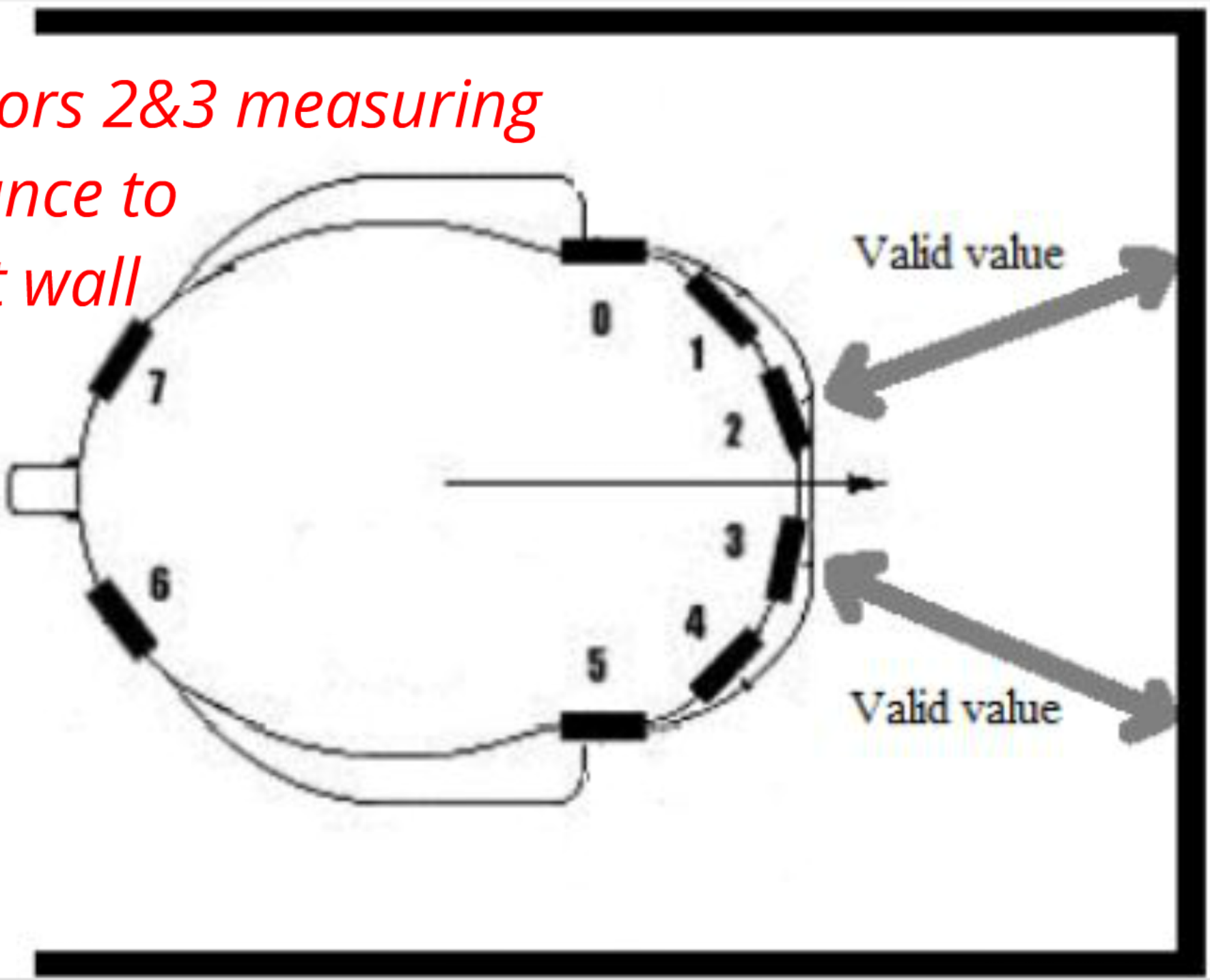


Minimum value of Sensors 0&1 or 4&5 measuring distance to wall




Sensors 0&5 detecting need for adjustment
> OT_THR
0
1
2
3
4
5
6
7
> OT_THR

Sensors 2&3 measuring distance to front wall

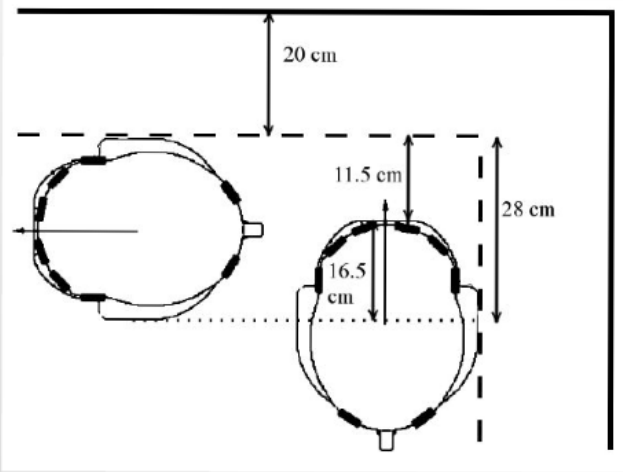


Navigating Corners

Outside Corner

- Loss of contact with parallel wall → adjust inward state
- Lateral sensor data "capped" at 0x0140 to prevent sharp turning
- Execute turn by adjusting inward until parallel to wall

Inside Corner



- Begin turn once front sensor reaches IT_THR threshold
- Inside turn state

Executing a Turn

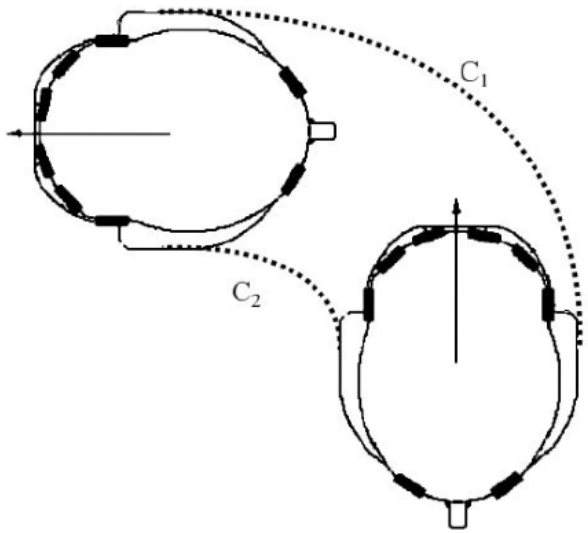
To turn a precise angle, use the following equation:

$$C_1 - C_2 = \theta R$$

- Robot has wheel track R = 24 cm

For Amigobot:

$$C_1 - C_2 \geq 0.5\pi(24) = 37.7 \text{ cm}$$



- 181 (0x00B5) ticks for quarter turn
- Due to consistent wheel slip, working constant 154 (0x009A)

Adjustment States

Once lateral sensors detect robot outside of ±1cm range of 20cm, robot enters one of two adjustment states:

- Robot feeds back detected range difference to left and right wheel velocities
- Cumulative adjustment → each loop iteration paused for 1 ms

$$\omega_i[n] = \omega_i[n-1] + k_1(R_1 - 20)$$
$$\omega_o[n] = \omega_o[n-1] - k_2(R_1 - 20)$$

SCOMP Alterations

MULT

- Used built-in multiplier LPM_MULT
- Result is spread across registers MHI and MLO

MLO

- Moves contents of LO register into accumulator

MHI

- Moves contents of HI register into accumulator

DIV and DREM

- Used built-in divider module LPM_DIVIDE

SHIFT2 and SHIFT3

- Used built-in shifter module LPM_CLSHIFT
- SHIFT2 implements variable shifter
- SHIFT3 implements rotating shifter

7-Segment Display

Left wheel velocity

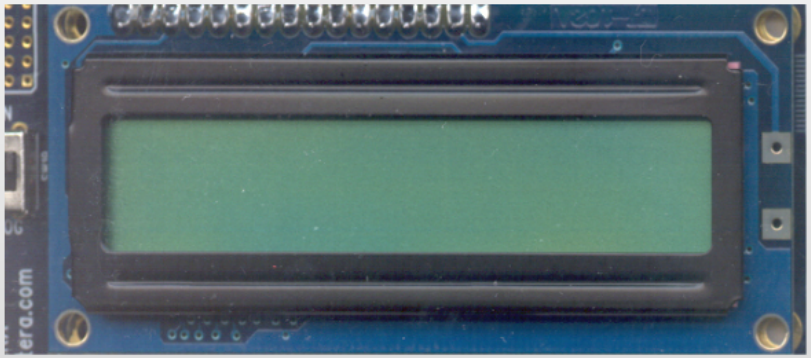
Average velocity

Right wheel velocity

Implementation

1. Add enables to IO Decoder
2. Add hex display modules and enable signal

LCD Display



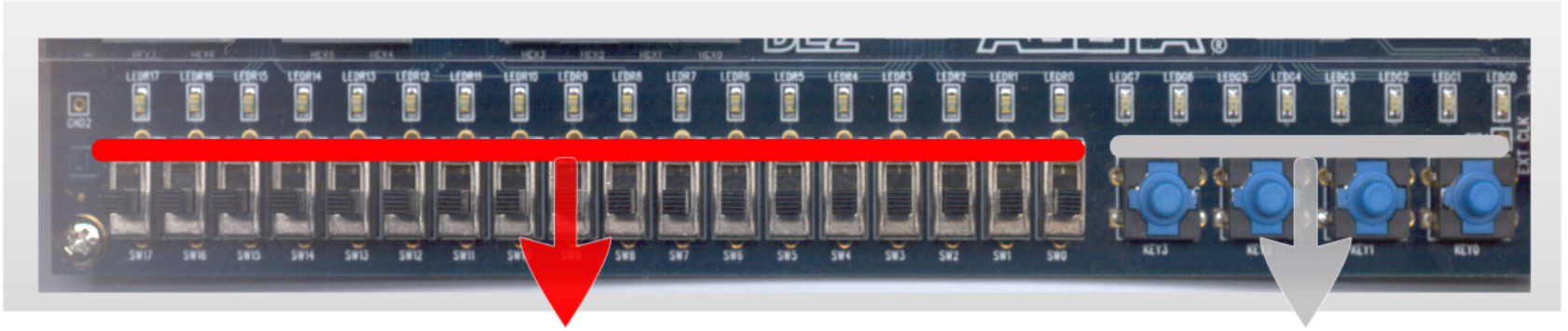
Implementation

1. Array of all ASCII values
2. Encode state in input

Shows state

- RDY!
- KEY2
- LEFT
- RGHT

LED Arrays



Red LEDs

Green LEDs

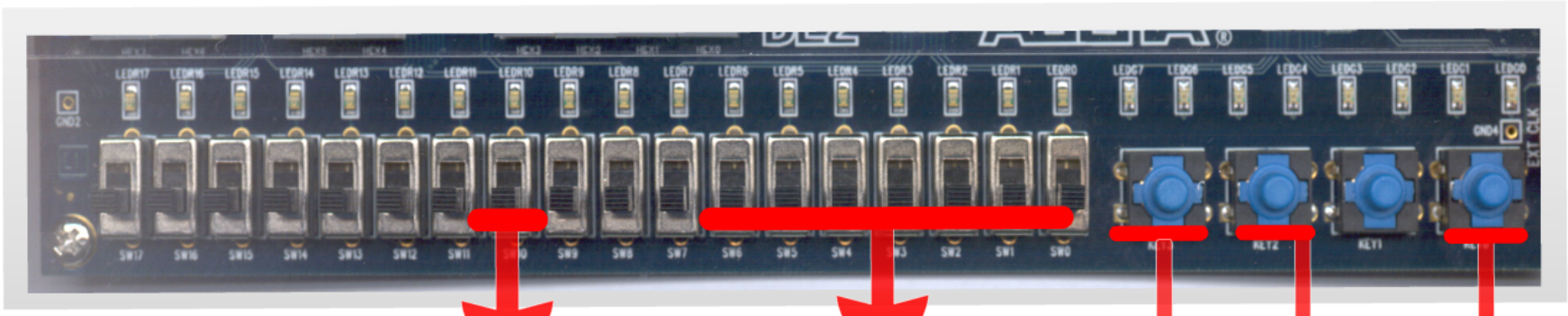
Implementation

1. Add enable to IO Decoder
2. Add LED module to supress

Use

1. Red LEDs turning rate
2. Green LEDs display loading bar animation

Assignments



Choose wall

Left = Down

Right = Up

Switch 10

Speed control

(binary)

Switches 0 - 6

Initialization

Key 3

Start

Key 2

Reset

Key 0

Problems Encountered

Nonfunctioning outside turn state →

Outside turning done by adjust inward state

Unreliable sensor readings →

Capped sensor readings

Reduce speed of robot

LCD display does not always change with states →

Ignored due to time constraints

Results

Percentage of course completed:

Right Wall: 100%

Left Wall: 100%

Collisions: 0

Accuracy Bonus: 5/10 sections

Time taken to complete course:

6 min, 15 sec

Discussion

Strengths

- Completes course
- Avoids collisions
- Speed control
- Informative display

Weaknesses

- Slow
- Inconsistent and inefficient algorithm

Proposed vs. Actual Design

LCD role changed

- Displays basic startup states instead of moving states

Outside turn state removed

- "Adjust in" state handles navigation of an outside corner
- Subroutine limits maximum value of lateral sensors to prevent the robot from turning too sharply

Pairs of sensors implemented

- Handles both lateral and forward wall sensing
- Lateral region R1 and forward region R2

Future Development

Changes

- Show states on LCD display
- Use green LEDs for states other than inside turn

Improvements to design

- Weighted values of sensors to detect robot orientation
- Improve accuracy of measurement algorithm to detect turns

What could be done differently

- Remove wait loop
- Change invalid value of sensors to 0x7FFF