Robot Racers

Gerry Chen, Faith Rodriguez, and Aditya Sridhar
Project Overview

In this project, two remote controlled robots race around a game space trying to run into checkpoints in order to earn points and to trigger power-ups.
Project Schematic

- Physical Control
  - Steering Wheel Pedal

- rFPGA (Receiver/Robot FPGA)
  - VGA module
  - Processor
  - UART FPGA communication
  - FPGA storage

- cFPGA (Controller FPGA)
  - ADC module
  - ADC storage
  - Processor
  - UART FPGA communication
  - FPGA storage

- Field Control

- Robots Checkpoints

Connections:
- (a) from Physical Control to rFPGA
- (b) from cFPGA to Robots Checkpoints
- (d) from rFPGA to Field Control
- (e) from Field Control to rFPGA

 Ardendend
Key Features

- Adopts a hardware-intensive approach
- Creates modularization of Verilog code into well-defined storage units to facilitate the flow of robot control
- Implements the five-stage pipelined processor with full bypassing from project checkpoint 4
- Incorporates Universal Asynchronous Receiver/Transmitter (UART) communication for multi-FPGA gaming
- Uses Arduino to control the robots and transmits and receives input using UART communication
High Level Design

- **Physical Control**
  - Steering wheel and pedal control the steering and throttle for each robot, respectively
  - Values are passed into an Analog-to-Digital Converter (ADC) to translate motion into a computable digital value

- **Controller FPGA**
  - Converts digital values received from the robots via the ADC and stores the values in an ADC storage (registers)
  - Uses UART communication to send and receive information about each robot’s state (speed/throttle and steering)
  - Handles speed/steering power-up duration logic for robots and stores motor effects

- **Receiver/Robot FPGA**
  - Determines and stores point values after robots touch a checkpoint
  - Outputs the robots’ points to the VGA
  - Uses UART communication to receive information about each robot’s state (points)

- **Field**
  - Contains robots and checkpoints that are controlled by the cFPGA
  - Robots and checkpoints interact with each other using Arduino
Building/Designing
<table>
<thead>
<tr>
<th>sender</th>
<th>receiver</th>
<th>command</th>
<th>payload</th>
<th>left motor vel</th>
<th>right motor vel</th>
</tr>
</thead>
<tbody>
<tr>
<td>cFPGA</td>
<td>robot</td>
<td>000</td>
<td>robot ID</td>
<td></td>
<td></td>
</tr>
<tr>
<td>robot</td>
<td>FPGAs</td>
<td>001</td>
<td>robot ID</td>
<td></td>
<td>touch duration</td>
</tr>
<tr>
<td>checkpoint</td>
<td>FPGAs</td>
<td>010</td>
<td>checkpoint ID</td>
<td></td>
<td>touch duration</td>
</tr>
<tr>
<td>rFPGA</td>
<td>cFPGA</td>
<td>100</td>
<td>game state</td>
<td></td>
<td>auxiliary info</td>
</tr>
<tr>
<td>cFPGA</td>
<td>rFPGA</td>
<td>101</td>
<td>powerup state</td>
<td></td>
<td>auxiliary info</td>
</tr>
</tbody>
</table>
Processor Logic

- Performs point-calculations for each robot
- Converts ADC digital values for speed/throttle and steering into left- and right-motor velocities
- Implements additional custom functions for reading from and writing to storage units:
  - Times and touch durations for robot-checkpoint touches found in a communication storage
  - Power-up time effect timeouts for robot-checkpoint touches found in a timeout storage
  - ADC digital values found in an ADC storage
  - Game time monitored during the game; used primarily in power-up effect calculation
- Instructions designed using MIPS Assembly
VGA for Robot Game

- A VGA display was created to keep track of the scores of both players
  - Images were converted to MIF files that were strategically chosen to be displayed
  - A decimal to binary converter interprets each value to choose which digit’s MIF file should be displayed
- Since point calculation was not a feature that we were able to complete, player 1’s score is simply a counter that aligns with game time in order to show that the VGA can in fact update in response to inputs
Snippet of cFPGA logic