ESE 501 Project 2 - Phase 1 – Robot navigation in confined Areas

*-Submitted by*

*Arnab Bhattacharya*

*Yash Jain*

**Design Summary**

**Robot**

In our simulation for this phase, we have 2 robots starting from 2 opposite corners of the map at the speed of 2m/s. The paths of the robots are independent and decided in advance by the server. The robots will move through the grids as indicated by the grid sequence. We implement the movement of the robots as synchronized with the positive edge of the clock. The robot moves across the 2m distance of the grid in 4 clock cycles thereby incrementing its position by 0.5m each cycle. Therefore, it sends the “CROSSING” signal at the third posedge to the server an waits for the “ACK” signal till the next posedge.

**Obstacle**

Similarly, we have 2 obstacles moving horizontally in the map at the speed of 4m/s. The obstacles also have a grid sequence and move according to that. The obstacles update their positions every cycles by 1m and cross each grid in 2 cycles. The obstacles move independently and are not controlled by the server.

**Server**

The server has the status of every robot and keeps updating them every cycle. Additionally, the server also stores the movement sequence for every robot. For this phase, we have only kept mutually independent movements for the robots. The server receives the “CROSSING” signal asynchronously. If the robot can move ahead and has no deadlock with other robots, then the server sends the “OK” signal for the robot to continue.

**Obstacle Detection**

We implement an integer value for every robot/obstacles that denotes its position inside the grid. For the detection of the obstacles, the robot queries the obstacle positions which are moving towards it. That is achieved by comparing the next grid values of the obstacle to its own next grid value to determine if the obstacle is moving towards it. If yes, then it proceeds to check the integer value determining its own position as well as the obstacles positions inside its own grid. Furthermore, these values will give us the distance between the robot and obstacle. If this distance comes about to be less than 3m, then the robot stops and waits for the distance to be greater than 3m. If the distance is greater than 3m, then the robot continues to move and checks the distance from the obstacle again at the next cycle.

By this way, the robots detect obstacles, avoid them if the distance is less than 3m and covers the path that they are supposed to travel as provided by the server before beginning their respective journeys.