ELEC 334 - Homework #1

Berat KIZILARMUT - 171024086

A. Problem 1

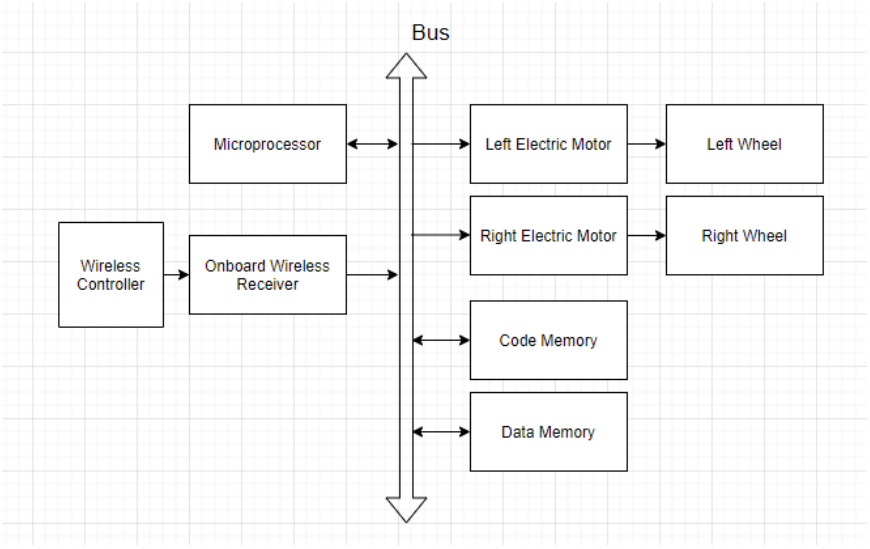

Figure 1

1. Hardware Block Diagram

Block diagram, Figure 1, has been designed for given remote-controlled 2-motor 3-wheel robot. Simplified the assumed components onboard to bare minimum.

Onboard microcontroller will have the necessary software, explained on the next part of the question, onboard and activate the electric motors accordingly.

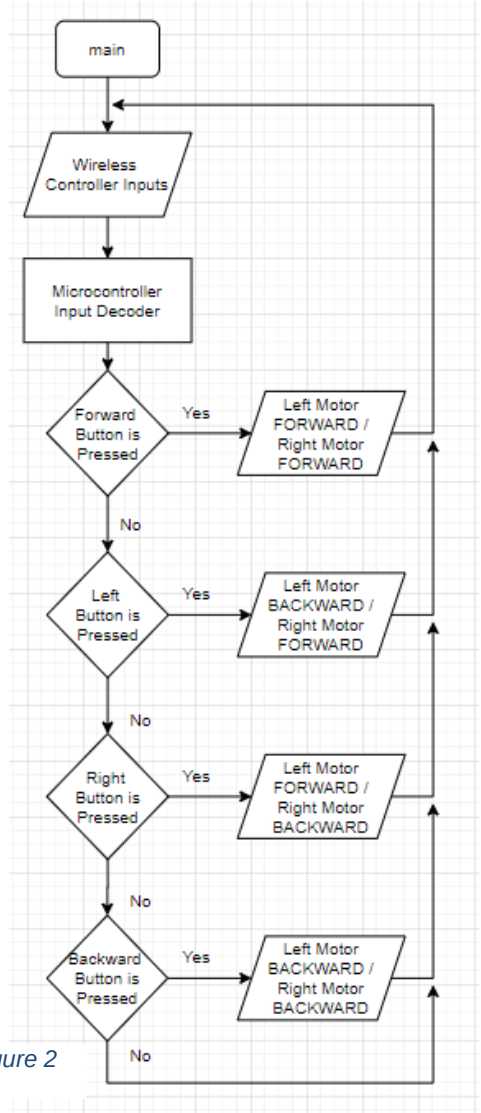

Figure 2

2. Software Flowchart

What happens in the said microcontroller on the previous stage is explained here. Motors on the robot are assumed to be able to work forwards and backwards.

Microcontroller checks the inputs coming from the wireless receiver. Depending on the input motors are activated forwards or backwards accordingly.

B. Problem 2

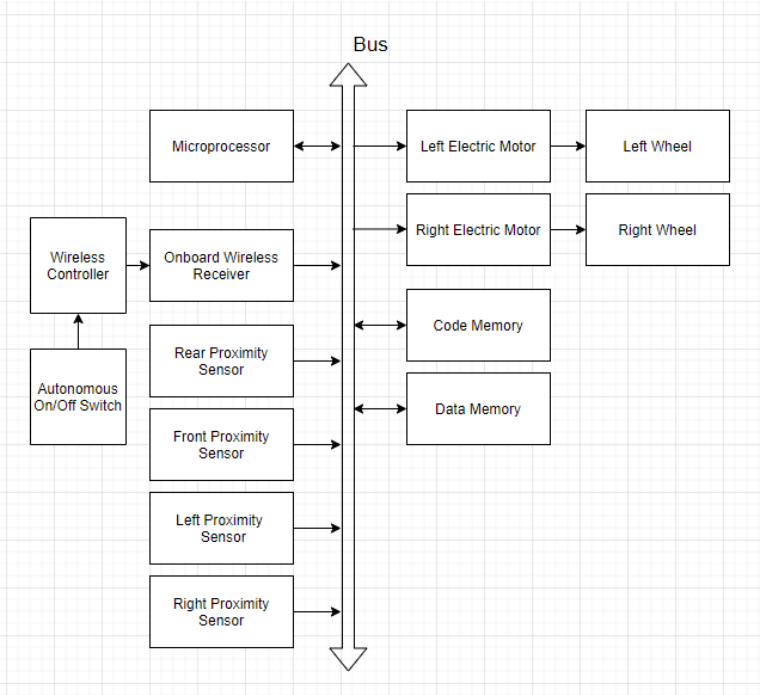

Figure 3

1.Hardware Block Diagram

Necessary sensors are added. These sensors report directly to the microcontroller. Microcontroller will ignore these sensors unless the “Autonomous Switch On” signal is received from the wireless receiver.

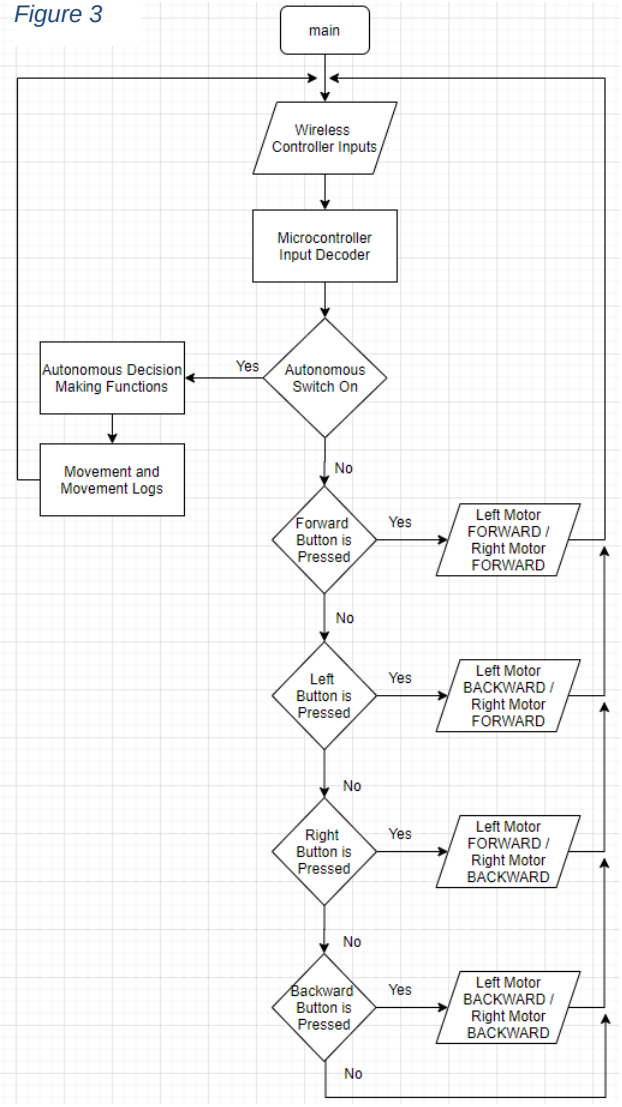

Figure 4

2. Software Flowchart

Autonomous decision functions could not be explained to the flowchart due to them making the flowchart too overcrowded.

Decisions will be made according to the sensor data, previous movement logs, has robot been there before, previous decisions on reoccurring positions etc.

C. Problem 3

Created a logger function using queue data structure method in C. This logger keeps and lists the previous and current locations, prints the list of moves.

```c
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
//Structure that represents list of previous paths
struct Pathlist
{
    int start, end, size, capacity;
    struct Node* list;
};
//Structure that stores the coordinate data
struct Node
{
    char x,y;
};
//Creates and starts a Pathlist with given capacity
struct Pathlist* startPathlist(int capacity)
{
    struct Pathlist* Pathlist = (struct Pathlist*) malloc(sizeof(struct Pathlist));
    Pathlist->start = 0;
    Pathlist->size = 0;
    Pathlist->capacity = capacity;
    Pathlist->end = capacity - 1;
    Pathlist->list = (struct Node*) malloc(sizeof(struct Node)*Pathlist->capacity);
    return Pathlist;
}
//Adds the given coordinate to the list in order
void push(struct Pathlist* Pathlist, struct Node node1)
{
    Pathlist->end = (Pathlist->end + 1) % Pathlist->capacity;
    Pathlist->list[Pathlist->end].x = node1.x;
    Pathlist->list[Pathlist->end].y = node1.y;
    Pathlist->size = Pathlist->size + 1;
}
//Prints the Pathlist
void printlist(struct Pathlist* Pathlist)
{
    for(int i=0; i<Pathlist->size; i++)
    {
        printf("%c %c\n", Pathlist->list[i].x, Pathlist->list[i].y);
    }
    printf("Number of previous locations: %d",Pathlist->size);
}
int main()
{
    //Creating a Pathlist with capacity of 20
    struct Pathlist* Pathlist = startPathlist(20);
    //Creating test string of coordinates
```

```
    struct Node coordinate1;
    coordinate1.x = 'E';
    coordinate1.y = '5';
    struct Node coordinate2;
    coordinate2.x = 'E';
    coordinate2.y = '6';
    struct Node coordinate3;
    coordinate3.x = 'D';
    coordinate3.y = '6';
    struct Node coordinate4;
    coordinate4.x = 'C';
    coordinate4.y = '6';
    struct Node coordinate5;
    coordinate5.x = 'B';
    coordinate5.y = '6';
        struct Node coordinate6;
    coordinate6.x = 'B';
    coordinate6.y = '7';
    //Pushing the created coordinates
    push(Pathlist, coordinate1);
    push(Pathlist, coordinate2);
    push(Pathlist, coordinate3);
    push(Pathlist, coordinate4);
    push(Pathlist, coordinate5);
    push(Pathlist, coordinate6);
    //Printing the full list requested
    printlist(Pathlist);
    //I've put this here because otherwise cmd prompt closes instantly
    int test;
    scanf("%d",test);
}
```

D. References

1. Queue (Introduction and Array Implementation), Geeksforgeeks, https://www.geeksforgeeks.org/queue-set-1introduction-and-array-implementation/
2. Diagram Making Tools, https://app.diagrams.net