GTU Department of Computer Engineering
CSE 433 – Spring 2023
Homework #2 Report

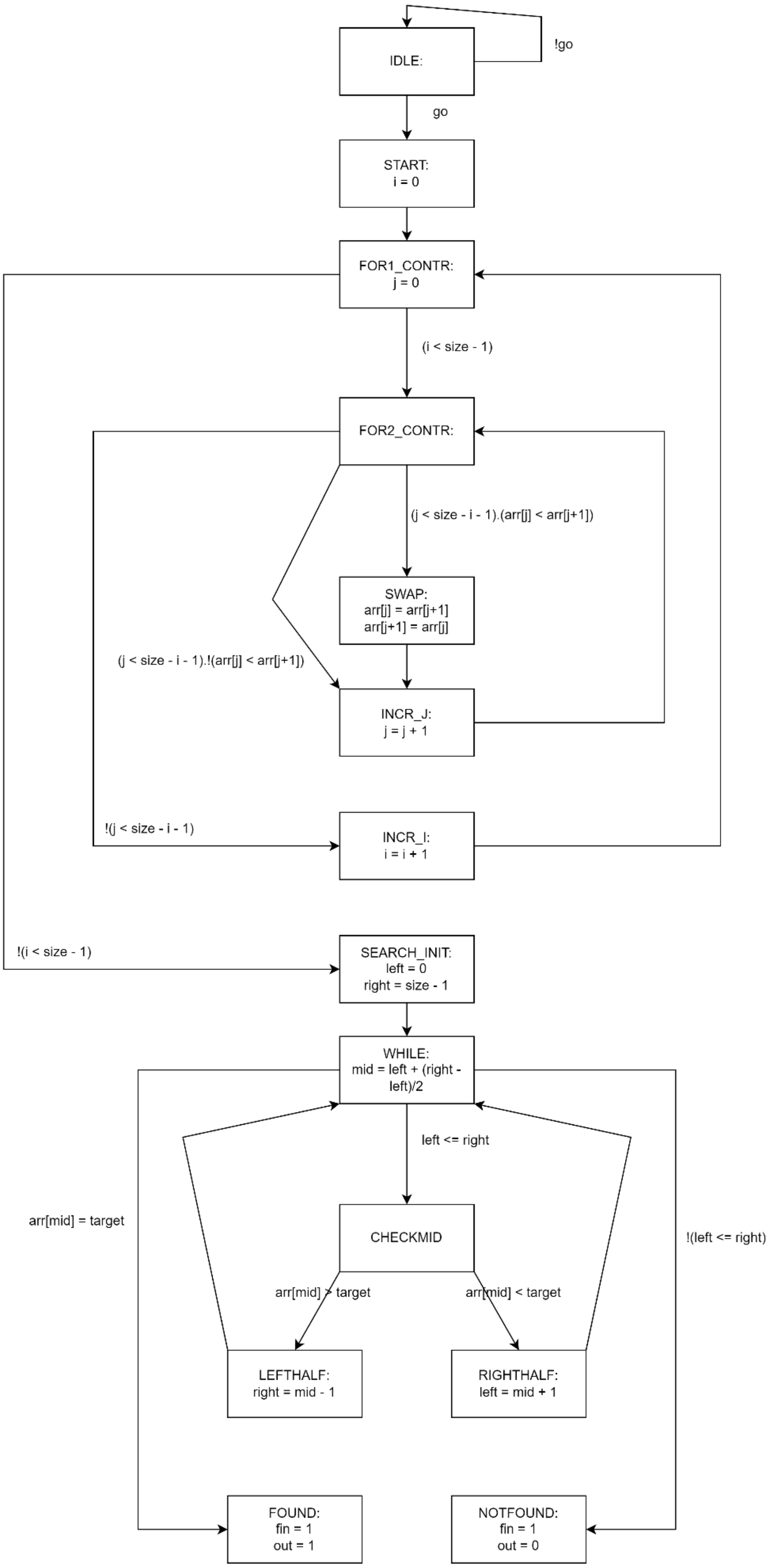
Hasan Mutlu
1801042673

C Code of Sortsearch

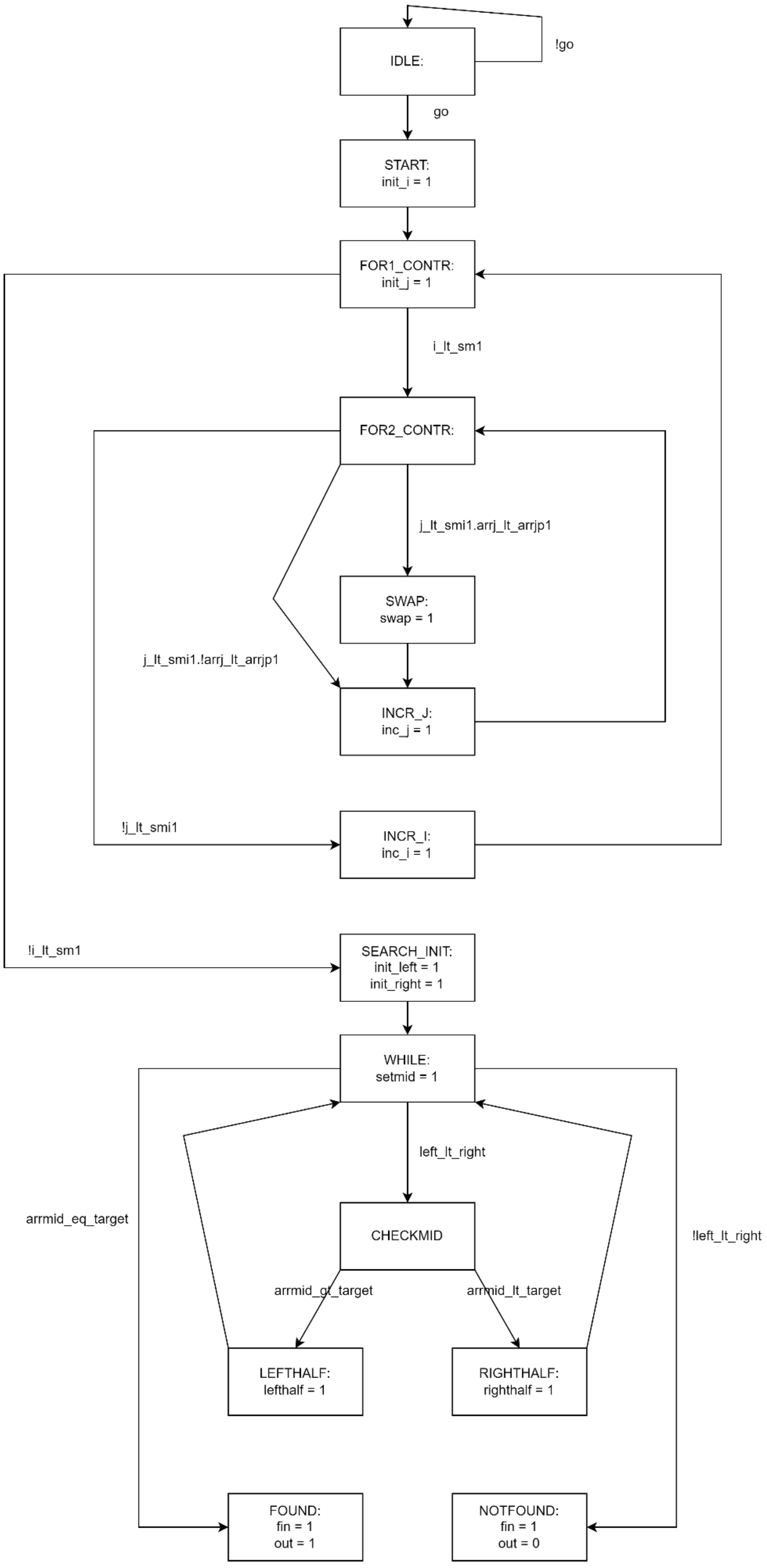
```c
int sortsearch(int arr[], int size, int target) {
    int i, j, temp, go;

    while(!go);

    i = 0;
    while ( i < size - 1) {
        j = 0;
        while(j < size - i - 1) {
            if (arr[j] > arr[j + 1]) {
                temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
            j++;
        }
        i++;
    }


    int left = 0;
    int right = size - 1;

    while (left <= right) {
        int mid = left + (right - left) / 2;

        if (arr[mid] == target) {
            return 1;
        }
        if (arr[mid] < target) {
            left = mid + 1;
        }
        else {
            right = mid - 1;
        }
    }

    return 0;
}
```

- ➔ Bubblesort algorithm is used for sorting.

Finite State Machine



Finite State Machine with Control Input-Outputs



Verilog Design and Assumptions

- ➔ In verilog design, the circuit takes the array as 32 different 32bit inputs. Giving the input as 2D memory was not possible.
- ➔ *“declaring module ports or function arguments with unpacked array types requires SystemVerilog extensions”*
- ➔ The datapath component, however, takes the entire array as a single concatenated 1024bit input and parses it into 32 diverse 32bit registers. It was possible to make the main component take the array as 1024bit input but it would make the testing harder.
- ➔ It is assumed that the user knows the array can be at most 32 elements long and gives valid inputs.

Test Results

In testbench, there is an array of 16 elements as input. Firstly, a number that is present in the array is searched. After the result is found, the machine is reset and an element that is not present in the array is searched.

The array:

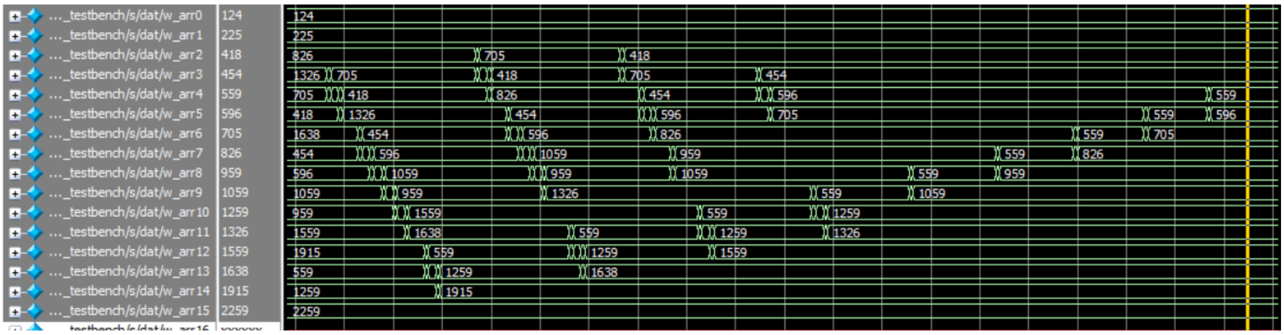
124, 1915, 225, 826, 1326, 705, 418, 1638, 454, 596, 1059, 959, 1559, 2259, 559, 1259;

First searched number: 418

Second searched number: 200

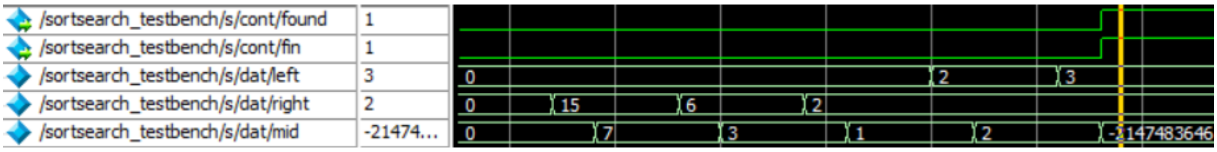
Searching 418

Sorting Result:



```
…_testbench/s/dat/w_arr0   124
…_testbench/s/dat/w_arr1   225
…_testbench/s/dat/w_arr2   418
…_testbench/s/dat/w_arr3   454
…_testbench/s/dat/w_arr4   559
…_testbench/s/dat/w_arr5   596
…_testbench/s/dat/w_arr6   705
…_testbench/s/dat/w_arr7   826
…_testbench/s/dat/w_arr8   959
…_testbench/s/dat/w_arr9   1059
…_testbench/s/dat/w_arr10  1259
…_testbench/s/dat/w_arr11  1326
…_testbench/s/dat/w_arr12  1559
…_testbench/s/dat/w_arr13  1638
…_testbench/s/dat/w_arr14  1915
…_testbench/s/dat/w_arr15  2259
```

Searching Process:

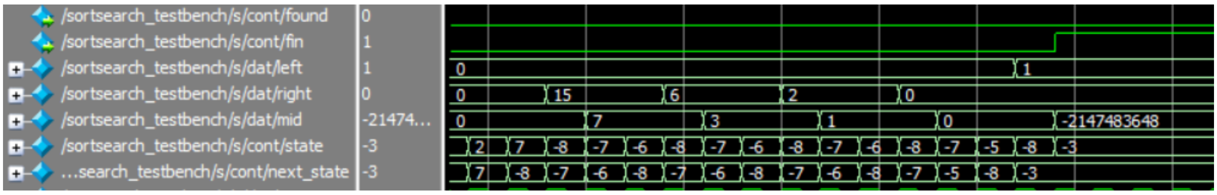


```
/sortsearch_testbench/s/cont/found  1
/sortsearch_testbench/s/cont/fin    1
/sortsearch_testbench/s/dat/left    3
/sortsearch_testbench/s/dat/right   2
/sortsearch_testbench/s/dat/mid     -21474...
```

Left and right are endpoints and the mid is the current middle index. Last value of mid is 2 and 418 is on arr[2]. The number is found in 3 iterations. Finish and found outputs are set to 1.

Searching 200

The array is sorted in the same manner.



```
/sortsearch_testbench/s/cont/found        0
/sortsearch_testbench/s/cont/fin          1
/sortsearch_testbench/s/dat/left          1
/sortsearch_testbench/s/dat/right         0
/sortsearch_testbench/s/dat/mid           -21474...
/sortsearch_testbench/s/cont/state        -3
...search_testbench/s/cont/next_state     -3
```

After searching in 4 iterations, the number is not found. Finish is set to 1 and found stays as 0.