CSE 433 Embedded Systems Project #2 Report Gökçe Nur Erer – 171044079

In this project it was requested to design a number analyzer FSM to find if a number is even/odd, a palindrome and is a Fibonacci number. To accomplish this FSM design, 4 different FSMs are implemented. Three FSM's check if a number is even/odd, a palindrome or not, a Fibonacci number or not, and the final FSM uses these FSM's to analyze a given number.

Even/Odd Number Checking FSM

In this FSM the design is based on the last bit of the given number. If the given number's last bit is 1 that means the number is odd, if it is 0 that means the number is even.

In this FSM 5 states are used:

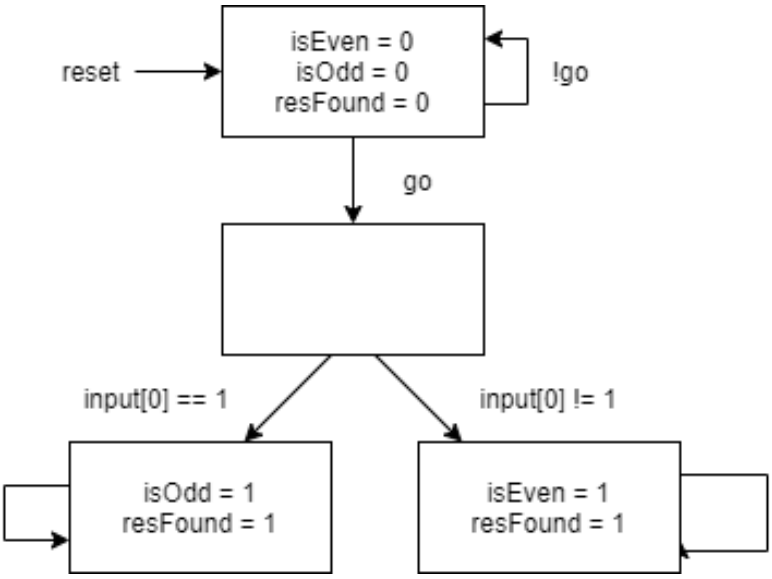
State 0: Variables isEven, isOdd and resFound are initialized as 0.

State 1: Checks if the last bit of the number is 1 or 0.

State 2: If state 1's check results true isOdd becomes 1 and resFound becomes 1.

State 3: If state 1's check results false isEven becomes 1 and resFound becomes 1.

FSM Diagram for Even/Odd Checking



Palindrome Checking FSM

In this FSM the design is based on the fact that if the number is written and calculated from backwards it would result as the same with the original number.

To show this clearer:

For example number 515 is a palindrome because:
Digit = 5
Sum = 5
Digit = 1
Sum = 5 * 10 + 1 = 51
Digit = 5
Sum = 51 * 10 + 5 = 515

But for example 355 is not a palindrome because:
Digit = 5
Sum = 5
Digit = 5
Sum = 5 * 10 + 5 = 55
Digit = 3
Sum = 55 * 10 + 3 = 558 which is not equal to 355

The corresponding C code to this solution is below:

```
int num,digit,reversedCalculated=0,temp;
int isPalindrome = 0;
temp=num;
while(temp>0)
{
    digit=temp%10;
    reversedCalculated=(reversedCalculated*10)+digit;
    temp=temp/10;
}
if(temp==reversedCalculated)
    isPalindrome = 1;
```

For this FSM 6 states are used:

State 0: Variable temp_num is initialized with the input number's value and the variable isPalindrome and resFound is initialized with 0.

State 1: Checks if temp_num is greater than 0.

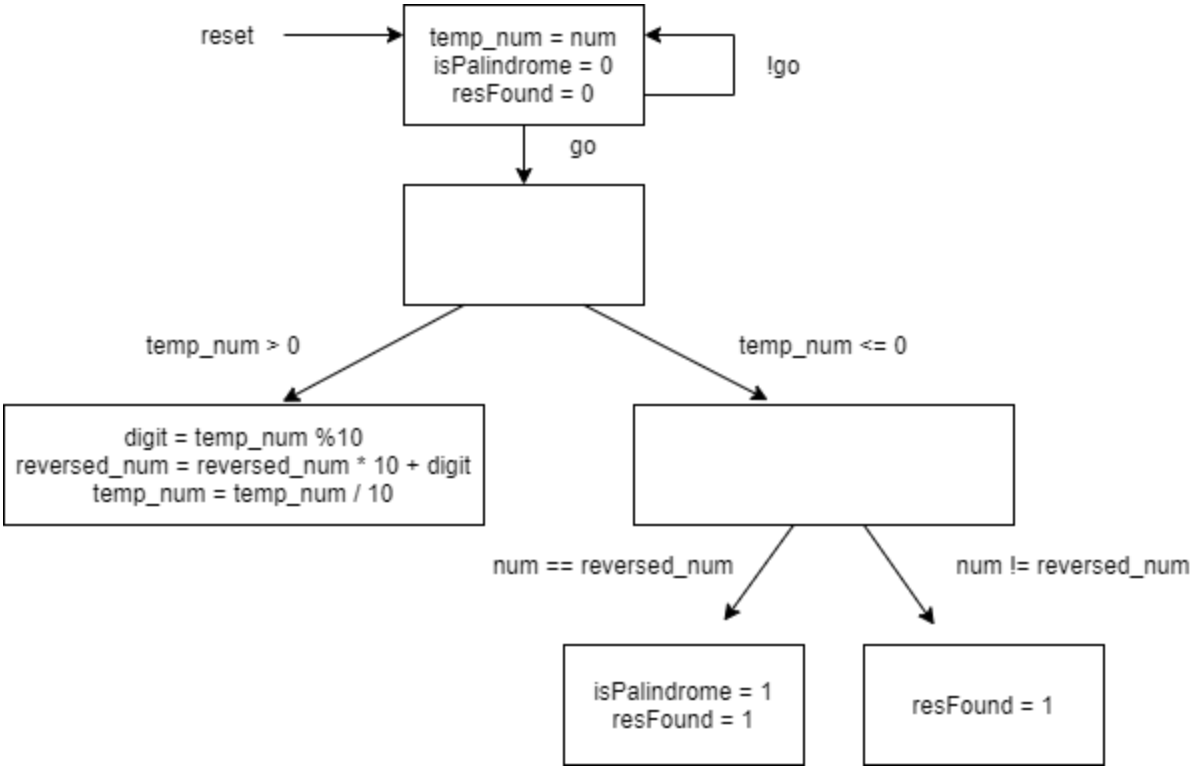
State 2: If state 1's check is true. Temp_num's modulo by 10 is taken and assigned to the digit variable. Then the reversed number calculation is done by multiplying the current reversed number calculation with 10 and adding the digit variable which is just calculated. After that temp_num is divided by 10.

State 3: If state 2's check is false. This state checks if the reversed number calculation is equal to our original number.

State 4: If state 3's check is true, that means the number is a palindrome so variables isPalindrome and resFound becomes 1.

State 5: If state 4's check is false, that means the number is not a palindrome so only the variable resFound becomes 1 and isPalindrome stays 0 since it was initialized at state 0.

FSM Diagram for Palindrome Checking



Fibonacci Checking FSM

In this FSM the design is based on calculating Fibonacci numbers as well as comparing calculated numbers with the input number. If the calculated Fibonacci number is equal to the input number, the input number is a Fibonacci number. If the calculated Fibonacci number is greater than the input number, the input number is not a Fibonacci number.

In this FSM 6 states are used:

State 0: In this state the variable resFound is initialized as 0.

State 1: In this state the variables prev and present are initialized as 1 for further calculation of the Fibonacci numbers.

State 2: In this state it is checked if the present Fibonacci number is smaller than the input number. Because if its smaller, more Fibonacci numbers has to be calculated until one of them becomes equal or greater than the input number.

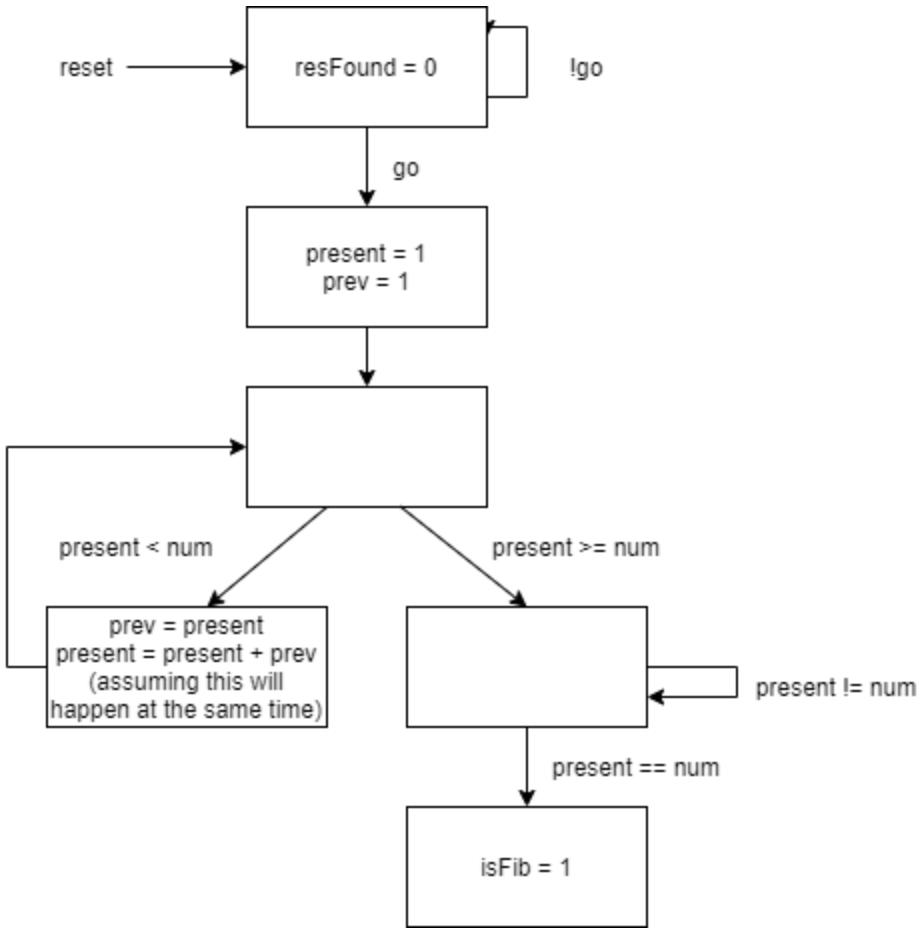
State 3: If the check in state 2 results true this state is run. In this state prev variable gets the value of the present. And present gets the value of present + prev. So the new Fibonacci number is calculated.

State 4: If the check in state 2 results false this state is run. In this state it is checked if the current Fibonacci number calculated is equal to the input number.

State 5: If the check in state 4 is true this state is run. In this state the value of isFib becomes 1 since now the two numbers are equal and that means the input number is a Fibonacci number.

PS: If the check in state 4 is false next state doesn't change, state 4 is run again until the condition is met.

FSM Diagram for Fibonacci Checking



Number Analyzer FSM

In this FSM all the mentioned FSM's above are used as modules and a single clock is given to all of the modules to enable parallel working of all the FSMs. Each FSM contains a resFound value within their module so this FSM uses this value to work.

This FSM has 5 state:

State 0: Initializes resFound with 0 and initalizes go_sig with the input go value. (This is done because after calculations we want all the modules to stop working at the last state of this FSM go_sig becomes 0 so FSMs don't keep on running.)

State 1: Checks if the module for the even/odd checking found a result which means foundEvenOdd variable is equal to 1.

State 2: If the check in state 1 is true this state is run. This state checks if the module for Fibonacci number checking found a result which means foundFibo variable is equal to 1.

PS: If the check in state 1 is false next state doesn't change, state 1 is run again until the condition is met.

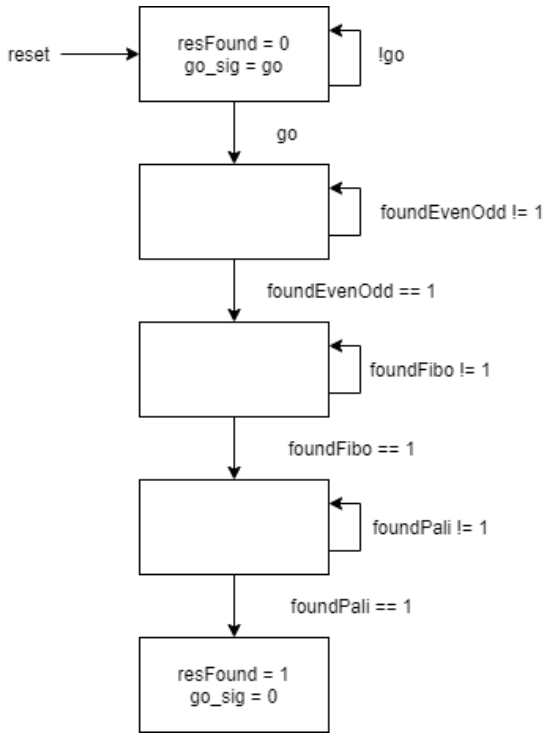
State 3: If the check in state 2 is true this state is run. This state checks if the module for palindrome checking found a result which means foundPali variable is equal to 1.

PS: If the check in state 2 is false next state doesn't change, state 2 is run again until the condition is met.

State 4: If the check in state 3 is true this state is run. This state changes the value of resFound to 1 since now all the modules have found a result and number is analyzed. And also to go_sig is 0 now since we don't want the modules to run over and over again.

PS: If the check in state 3 is false next state doesn't change, state 3 is run again until the condition is met.

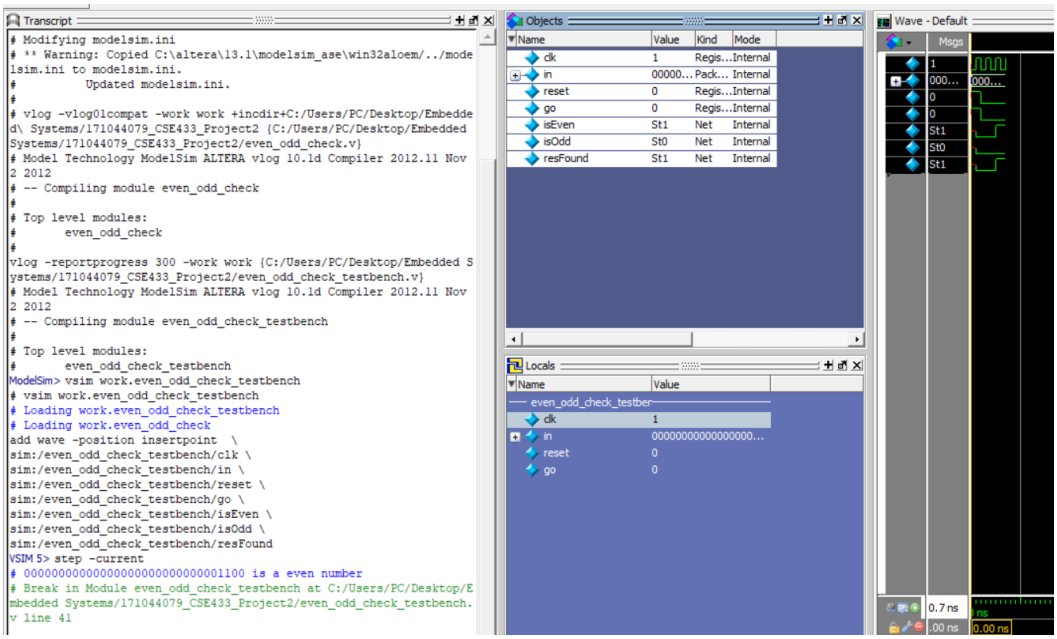
FSM Diagram for Number Analyzer



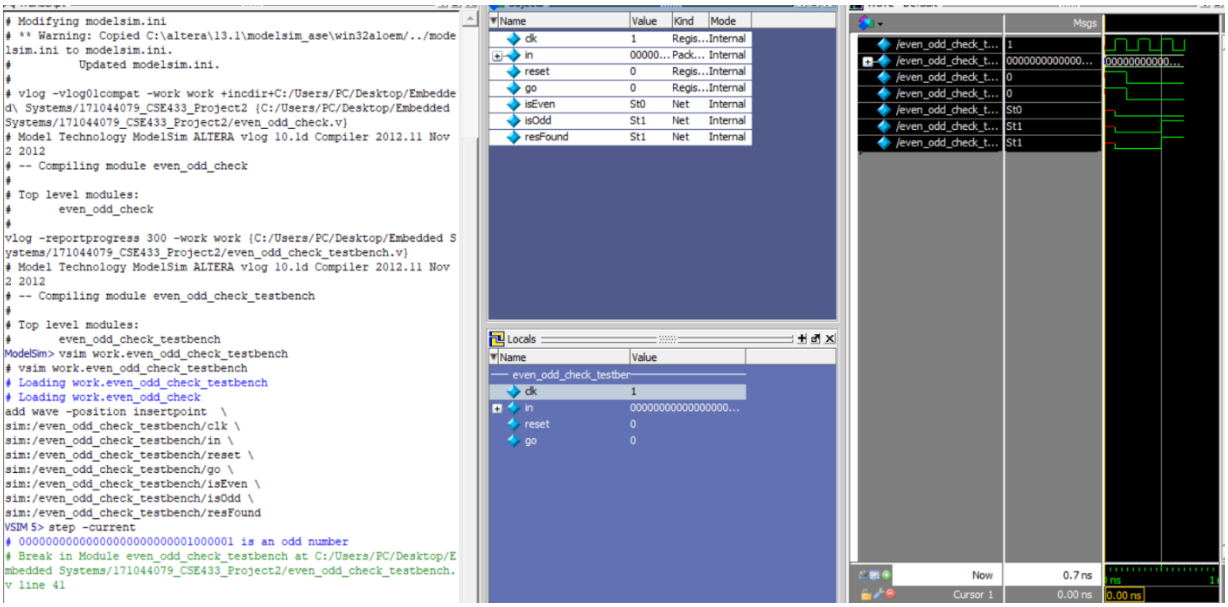
Results of individual FSMs and Number Analyzer

Results for Even/Odd Checking FSM

- **For Even Number : 12**

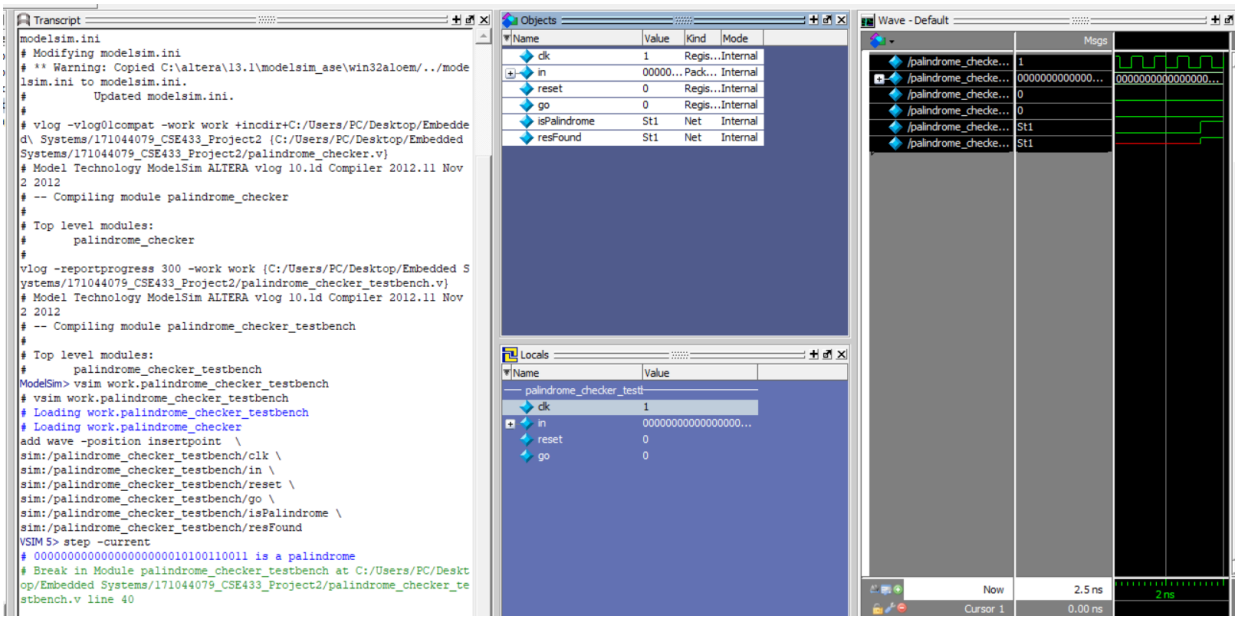


- **For Odd Number : 65**

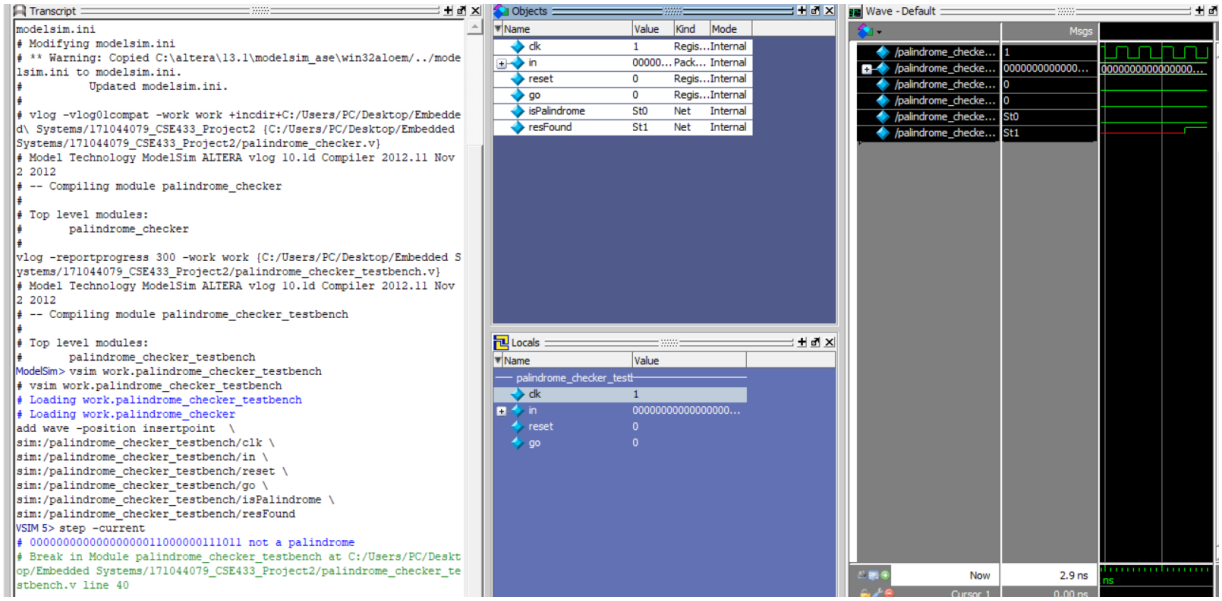


Results for Palindrome Checking FSM

- **For Palindrome Number : 1331**

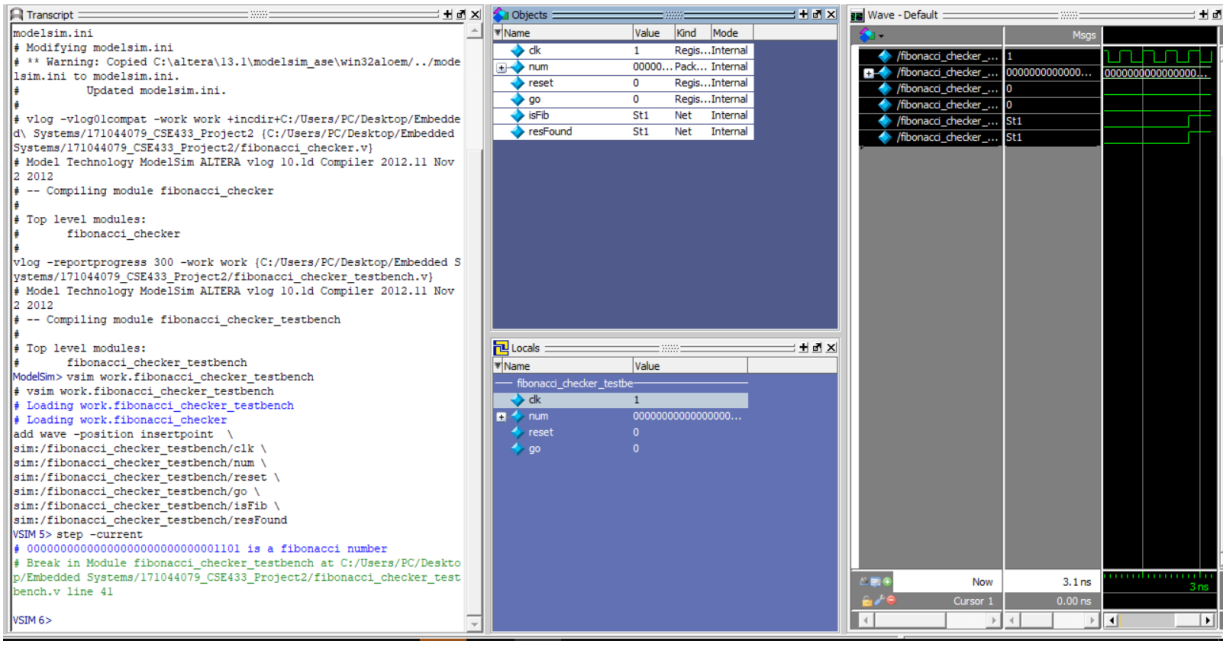


- **For Non-Palindrome Number : 12347**

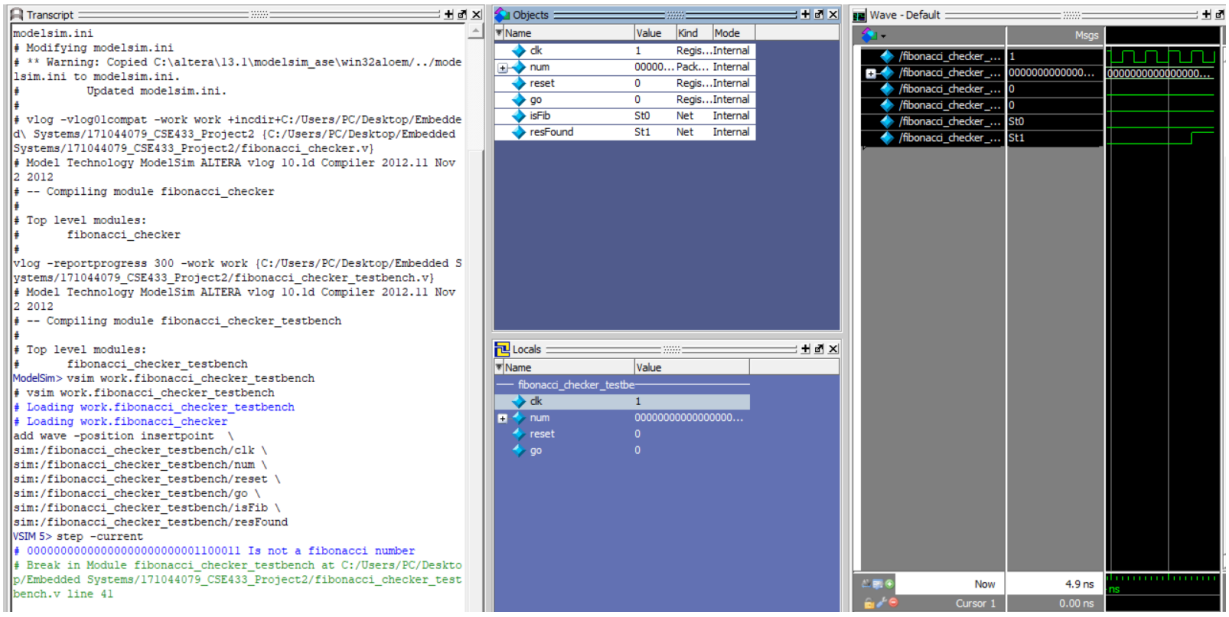


Results for Fibonacci Checking FSM

- **For Fibonacci Number : 13**

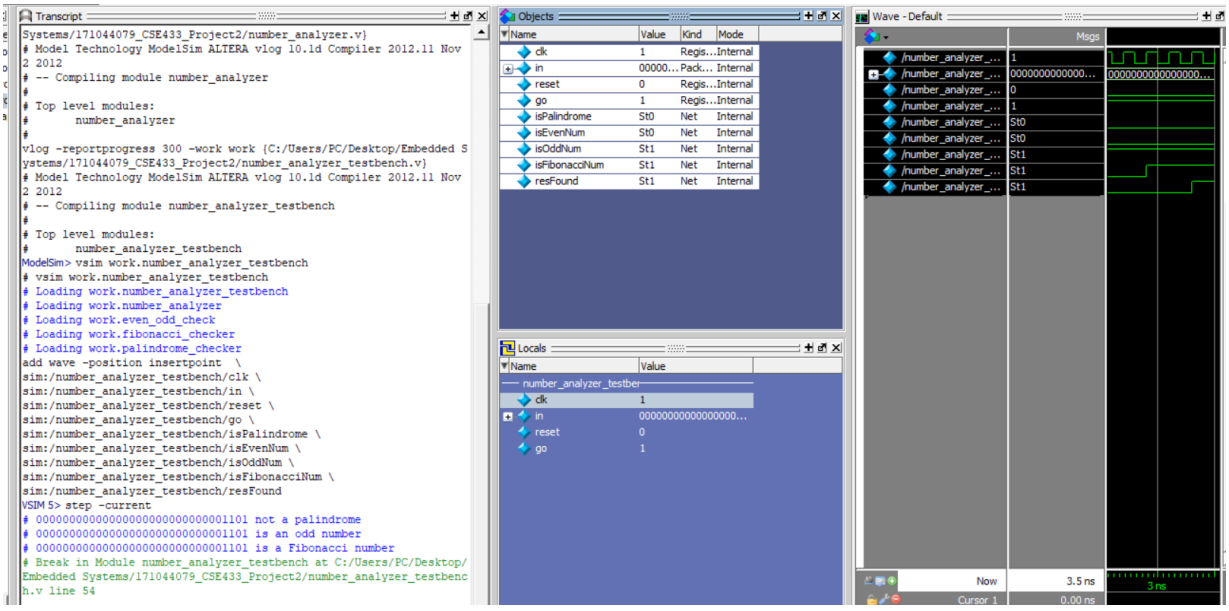


- **For Non-Fibonacci Number: 99**

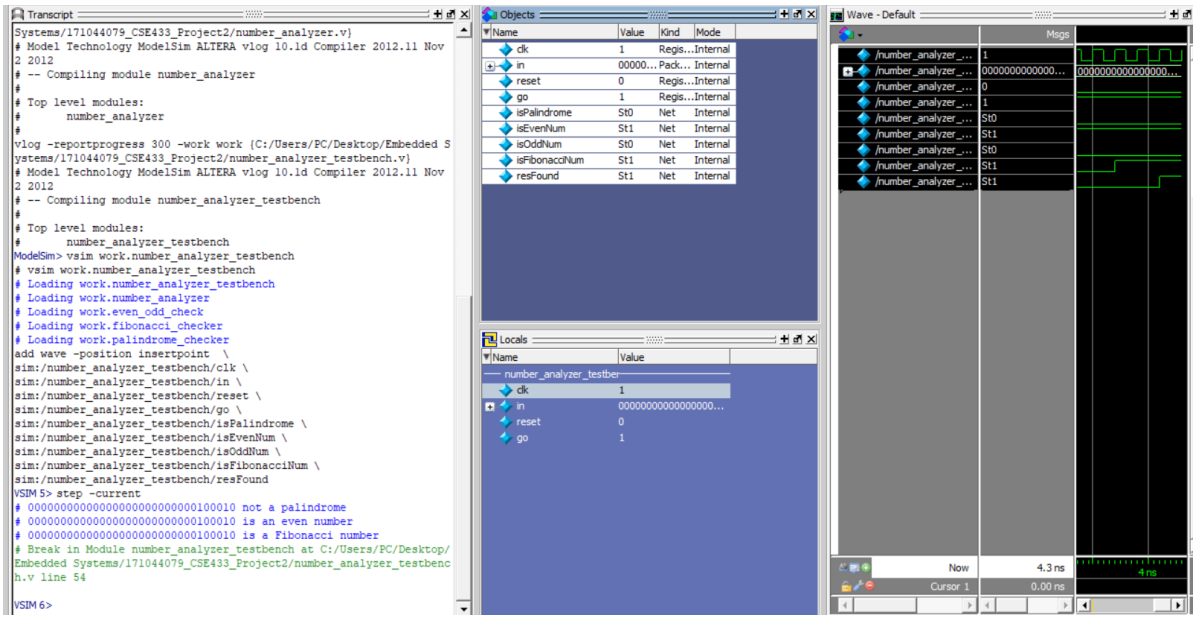


Results for Number Analyzer FSM

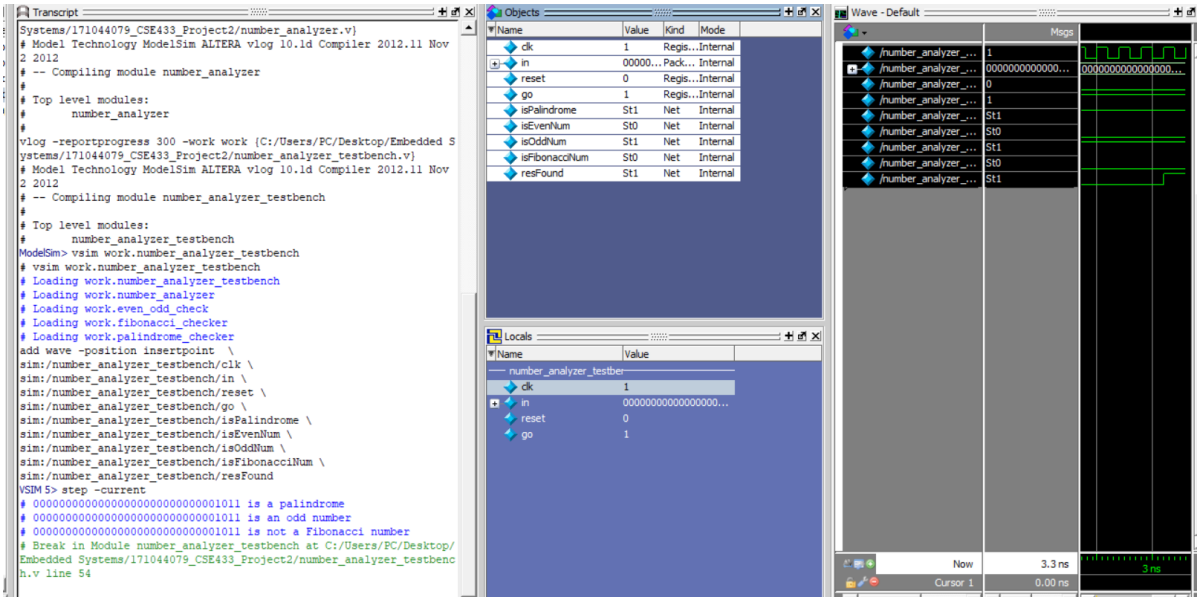
- **For Fibonacci, Non-Palindrome, Odd Number : 13**



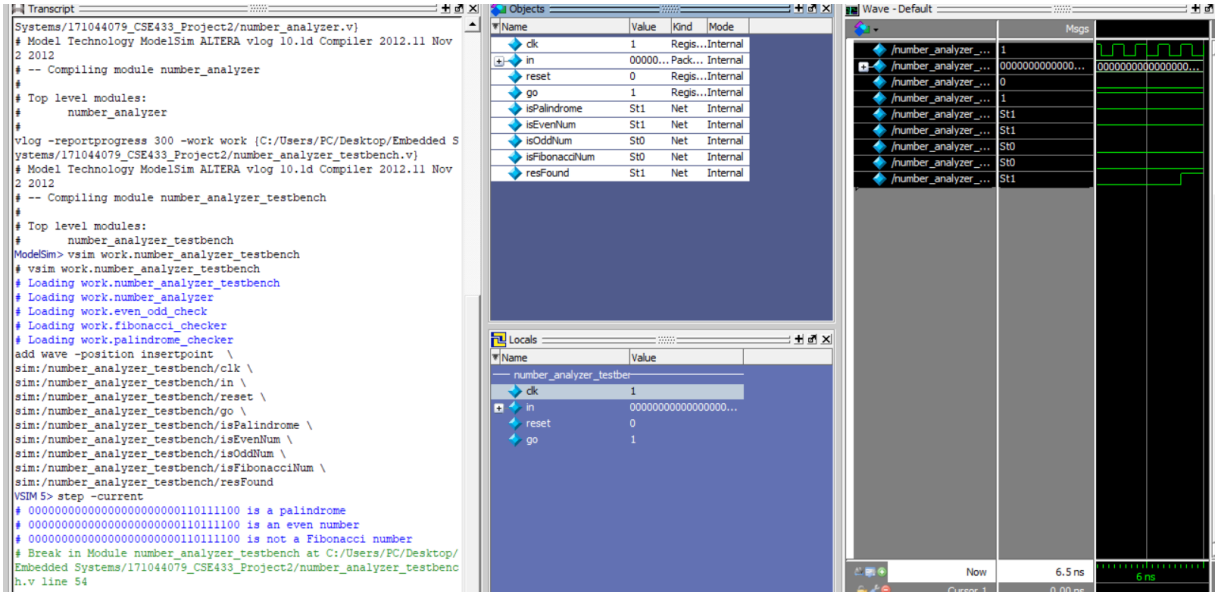
- **For Fibonacci, Non-Palindrome, Even Number: 34**



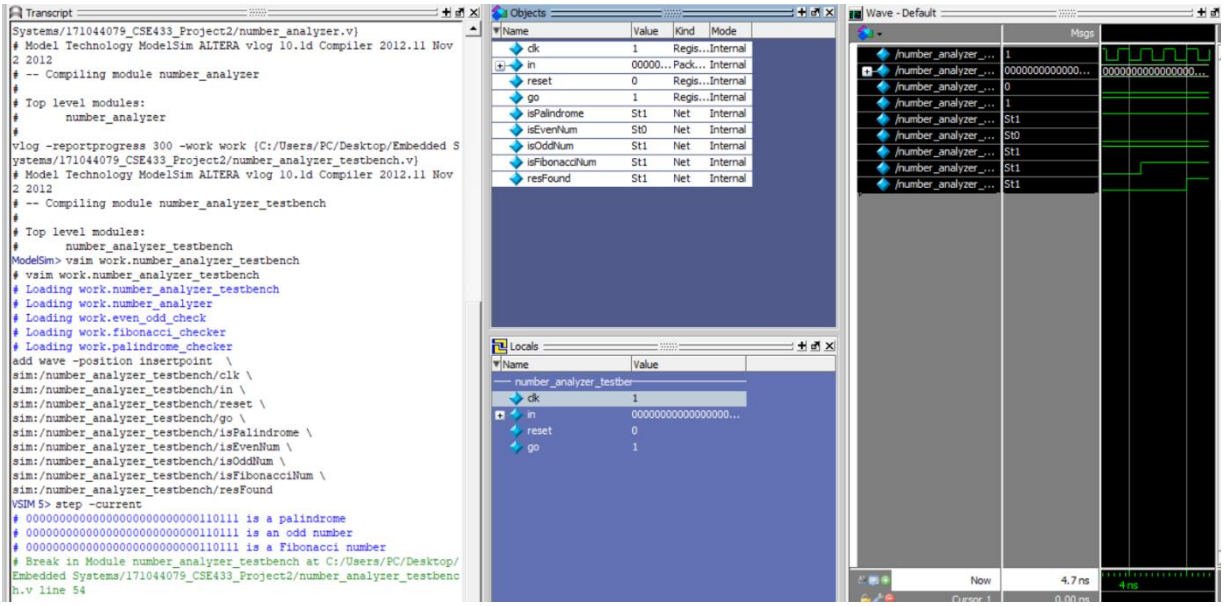
- **For Non-Fibonacci, Palindrome, Odd Number : 11**



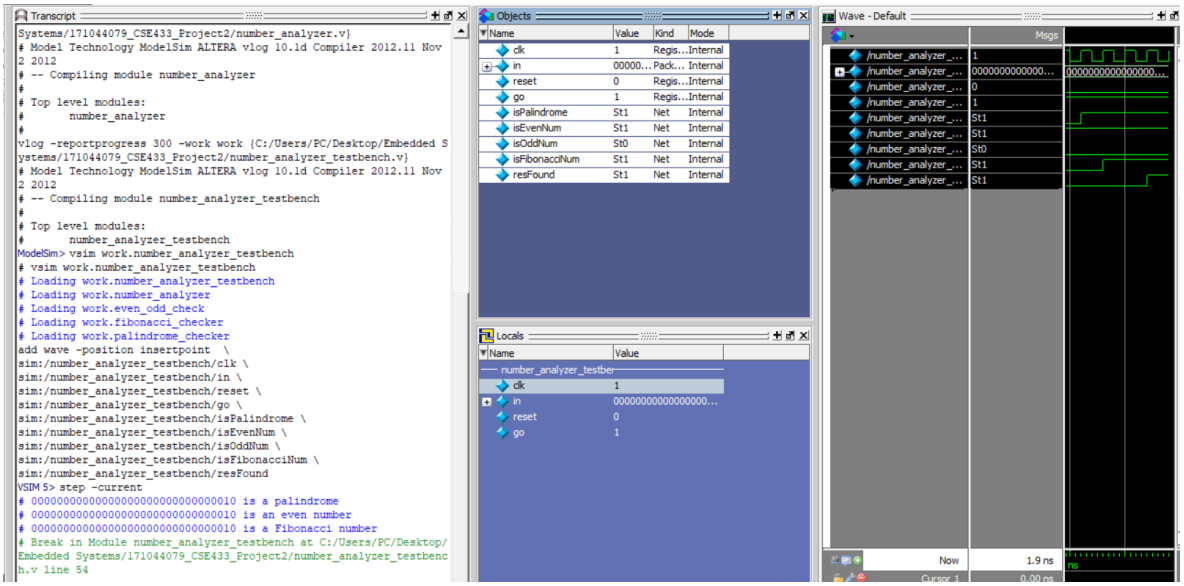
- **For Non-Fibonacci, Palindrome, Even Number: 444**



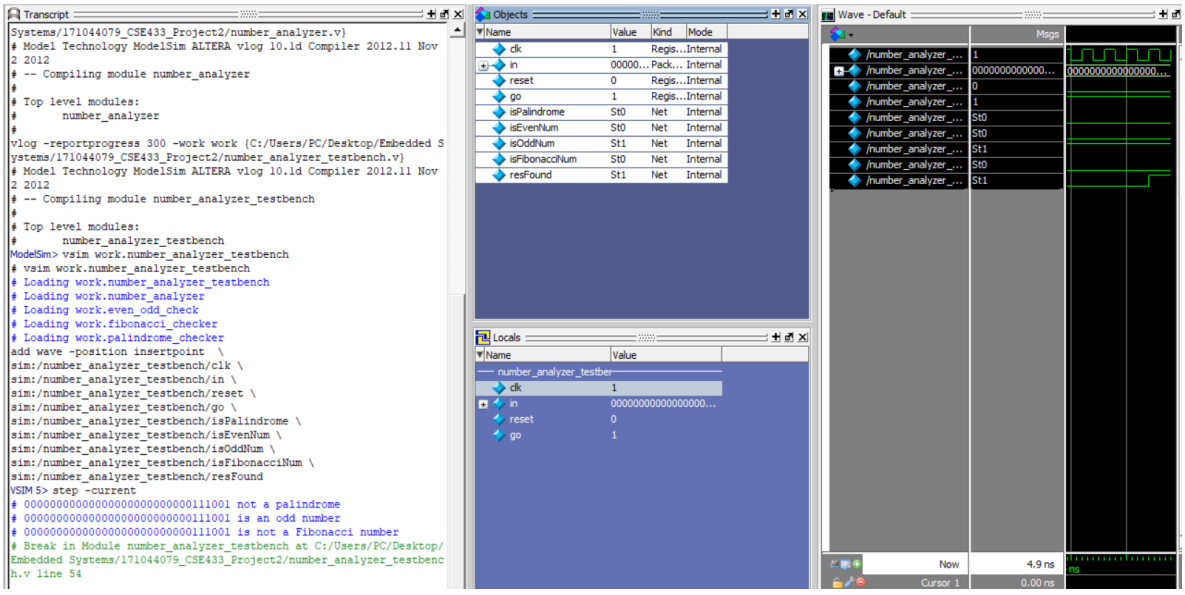
- **For Fibonacci, Palindrome, Odd Number: 55**



- **For Fibonacci, Palindrome, Even Number: 2**



- **For Non-Fibonacci, Non-Palindrome, Odd Number: 57**



- **For Non-Fibonacci, Non-Palindrome, Even Number: 8000**