Construction of a Basic Calculator through the Turing Machine – A Review
Ezhilarasu P
Associate Professor
Department of Computer Science and Engineering
Hindusthan College of Engineering and Technology, Coimbatore
Tamil Nadu – India

ABSTRACT
This paper focuses on the Turing Machine usage as a basic calculator. The basic calculator here refers the four primary arithmetic operations. Those are namely 1. Addition, 2. Subtraction, 3. Multiplication, and 4. Division. The Turing Machine has the favorable circumstances as compared with the other abstract machines the Finite Automata and the Pushdown Automata. The Turing Machine has the vital feature called bi-directional movement. This aspect makes the Turing Machine to support the construction of the basic calculator. The transition diagram and transition table also derived for each calculation. The processing of input for each operation also given.

Keywords:- Turing Machine, Addition, Subtraction, Multiplication, Division

I. INTRODUCTION
A. Turing Machine
The Turing Machine is septuplets. The Turing machine mathematically models a mechanically operating machine on a tape [1]. It has the tape for taking input from the cell. The tape by default has the blank symbol (#) in all the cells. The head for moving in the right or left direction to perform read/write operation. A state register that has the initial value, as a start state for storing the Turing Machine's state. Then a finite table of instruction.

B. Basic Calculator
The basic calculation frequently refers the basic arithmetic operations. Those are
1. Addition (represented by the operator ‘+’)
2. Subtraction (represented by the operator ‘-’)
3. Multiplication (represented by the operator ‘×’ or ‘·’ or ‘*’)
4. Division (represented by the operator ‘÷’ or ‘/’)

II. RELATED WORKS
[11]. Hopcraft J E et. al [2001] explained about the subroutine usage in the Turing Machine [12]. This subroutine technique was used to construct the Turing Machine for multiplication and division. Martin J C [2010] explained the Turing Machine to support the implementation of palindrome and the reversing the string [13]. Ezhilarasu P & Sowmya R [2015] constructed the Turing Machine to check the equivalence of the input string for the three input characters [14].

III. ARITHMETIC OPERATIONS
The Turing Machine used to perform the arithmetic operations namely addition, subtraction, multiplication, and division explained using the transition diagram, transition table, and the example input.

A. ADDITION
The Turing Machine that performs the addition operation as shown in figure 1, and table 1.

Fig: 1. Transition Diagram of the Turing Machine to perform the addition operation
Table. I. Transition Table of the Turing Machine that perform the addition operation

<table>
<thead>
<tr>
<th>States</th>
<th>Input</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>E</td>
</tr>
<tr>
<td>s₀</td>
<td>-</td>
</tr>
<tr>
<td>s₁</td>
<td>E, R, s₁</td>
</tr>
<tr>
<td>s₂</td>
<td>E, R, s₂</td>
</tr>
<tr>
<td>s₃</td>
<td>#, S, h₃</td>
</tr>
</tbody>
</table>

INPUT PROCESSING FOR THE ADDITION

The input given here is unary ‘E’. Ex. The number 5 represented as EEEEE.

Input: #EEEEEE+EE#

so, #EEEEEE+EE#

Step 1: s₁, #EEEEEE+EE#

Step 2: s₁, #EEEEEE+EE#

Step 3: s₁, #EEEEEE+EE#

Step 4: s₁, #EEEEEE+EE#

Step 5: s₁, #EEEEEE+EE#

Step 6: s₁, #EEEEEE+EE#

Step 7: s₂, #EEEEEE+EE#

Step 8: s₂, #EEEEEE+EE#

Step 9: s₂, #EEEEEE+EE#

Step 10: s₃, #EEEEEE+EE#

h₃, #EEEEEE+EE#

The Turing Machine reached the halting state for acceptance. The final input string is #EEEEEE+, i.e seven E’s. 5+2=7. Hence the construction of the Turing Machine is correct for addition operation.

B. SUBTRACTION

The Turing Machine that performs the subtraction operation as shown in figure 2, and table 2.

Table. II. Transition Table of the Turing Machine that perform the subtraction operation

<table>
<thead>
<tr>
<th>States</th>
<th>Input</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>E</td>
</tr>
<tr>
<td>s₀</td>
<td>-</td>
</tr>
<tr>
<td>s₁</td>
<td>#, R, s₂</td>
</tr>
<tr>
<td>s₂</td>
<td>E, R, s₁</td>
</tr>
<tr>
<td>s₃</td>
<td>E, R, s₁</td>
</tr>
<tr>
<td>s₄</td>
<td>#, L, s₃</td>
</tr>
<tr>
<td>s₅</td>
<td>E, L, s₆</td>
</tr>
<tr>
<td>s₆</td>
<td>E, L, s₆</td>
</tr>
<tr>
<td>s₇</td>
<td>E, L, s₇</td>
</tr>
<tr>
<td>s₈</td>
<td>E, L, s₈</td>
</tr>
</tbody>
</table>

INPUT PROCESSING FOR THE SUBTRACTION

In subtraction, we have three types of conditions. Those are 1. The two numbers are same, 2. The first operand(number) greater than the second operand(number), and 3. The second number higher than the first number. Hence, we need to check the three types of input. The input given here is unary ‘E’. Ex. The number 3 represented as EEE.

Input Type 1: #EE-EE#

s₀, #EEE-EE#
Step 1: $s_1,##\text{EE-EE}\#

Step 2: $s_2,##\text{EE-EE}\#

Step 3: $s_2,##\text{EE-EE}\#

Step 4: $s_3,##\text{EE-EE}\#

Step 5: $s_3,##\text{EE-EE}\#

Step 6: $s_3,##\text{EE-EE}\#

Step 7: $s_4,##\text{EE-EE}\#

Step 8: $s_5,##\text{EE-EE}\#

Step 9: $s_6,##\text{EE-EE}\#

Step 10: $s_7,##\text{EE-EE}\#

Step 11: $s_8,##\text{EE-EE}\#

Step 12: $s_1,##\text{EE-EE}\#

Step 13: $s_2,##\text{EE-EE}\#

Step 14: $s_3,##\text{EE-EE}\#

Step 15: $s_3,##\text{EE-EE}\#

Step 16: $s_4,##\text{EE-EE}\#

Step 17: $s_5,##\text{EE-EE}\#

Step 18: $h_8,##\text{EE-EE}\#

The Turing Machine reached the halting state for acceptance. The final input string is $##\text{EE-EE}\#$, i.e., one E’s. $2-1=1$. Hence, the construction of the Turing Machine is correct for the subtraction operation of input type 2.

Input Type 3: $\#\text{EE-EE}\#

S., $\#\text{EE-EE}\#

Step 1: $s_1,\#\text{EE-EE}\#

Step 2: $s_2,##\text{EE-EE}\#

Step 3: $s_3,##\text{EE-EE}\#

Step 4: $s_3,##\text{EE-EE}\#

Step 5: $s_4,##\text{EE-EE}\#

Step 6: $s_5,##\text{EE-EE}\#

Step 7: $s_6,##\text{EE-EE}\#

Step 8: $s_7,##\text{EE-EE}\#

Step 9: $s_8,##\text{EE-EE}\#

h_8,##\text{EE-EE}\#

The Turing Machine reached the halting state for acceptance. The final input string is $##\text{EE-EE}\#$, i.e minus followed by one E’s. $1-2=1$. Hence, the construction of the Turing Machine is correct for the subtraction operation of input type 3.

C.MULTIPLICATION

The Turing Machine that performs the multiplication operation as shown in figure 3.
The subroutine concept used to implement the multiplication operation [12]. The input given here is unary ‘1’. Ex. The number 7 represented as 1111111.

Input: #11*111####

s0, #11*111####

Step 1: s1, #1*111####
Step 2: s4, #1*111####
Step 3: s5, #1*111####
Step 4: s6, #1*111####
Step 5: s6, #1*111####
Step 6: s6, #1*111####
Step 7: s7, #1*111####
Step 8: s7, #1*111####
Step 9: s7, #1*111####
Step 10: s7, #1*111####
Step 11: s5, #1*111####
Step 12: s6, #1*111####
Step 13: s6, #1*111####
Step 14: s6, #1*111####
Step 15: s7, #1*111####
Step 16: s7, #1*111####
Step 17: s7, #1*111####
Step 18: s7, #1*111####
Step 19: s6, #1*111####
Step 20: s6, #1*111####
Step 21: s6, #1*111####
Step 22: s6, #1*111####
Step 23: s7, #1*111####
Step 24: s7, #1*111####
Step 25: s7, #1*111####
Step 26: s7, #1*111####
Step 27: s8, #1*111####
Step 28: s8, #1*111####
Step 29: s8, #1*111####
Step 30: s9, #1*111####
Step 31: s9, #1*111####
Step 32: s9, #1*111####
Step 33: s1, #1*111####
Step 34: s2, #1*111####
Step 35: s2, #1*111####
Step 36: s2, #1*111####
Step 37: s2, #1*111####
Step 38: s2, #1*111####
Step 39: s2, #1*111####
Step 40: s2, #1*111####
Step 41: s3, #1*111####

The Turing Machine reached the halting state for acceptance. The final input string is #11111111. i.e six 1’s. 2*3=6. Hence the construction of the Turing Machine is correct for the multiplication operation.

D. DIVISION

The Turing Machine that performs the division operation as shown in figure 4.
Step 5: s2, #1111-1E####, Step 6: s3, #1111-1E####

Step 7: s3, #1111-1E####, Step 8: s3, #1111-1E####

Step 9: s4, #1111-1E####, Step 10: s5, #1111-1E#####

Step 11: s8, #1111-1E#####, Step 12: s9, #1111-1E####

Step 13: s9, #1111-1E####, Step 14: s9, #1111-1E####

Step 15: s9, #1111-1E####, Step 16: s1, #1111-1E#####

Step 17: s2, #1111-1E####, Step 18: s2, #1111-1E####

Step 19: s2, #1111-1E####, Step 20: s3, #1111-1E####

Step 21: s3, #1111-1E####, Step 22: s3, #1111-1E####

Step 23: s4, #1111-1E####, Step 24: s4, #1111-1E####

Step 25: s5, #1111-1E#####, Step 26: s6, #1111-1E#####

Step 27: s6, #1111-1E#####, Step 28: s6, #1111-1E#####

Step 29: s8, #1111-11Z####, Step 30: s8, #1111-11Z####

Step 31: s8, #1111-11Z####, Step 32: s9, #1111-11Z####

Step 33: s9, #1111-11Z####, Step 34: s9, #1111-11Z####

Step 35: s1, #1111-11Z####, Step 36: s2, #1111-11Z####

Step 37: s2, #1111-11Z####, Step 38: s3, #1111-11Z####

Step 39: s3, #1111-11Z####, Step 40: s3, #1111-11Z####

Step 41: s4, #1111-11Z####, Step 42: s5, #1111-11EZ####

Step 43: s8, #1111-11EZ####, Step 44: s9, #1111-11EZ####

Step 45: s9, #1111-11EZ####, Step 46: s1, #1111-11EZ####

Step 47: s2, #1111-11EZ####, Step 48: s3, #1111-11EZ####

Step 49: s3, #1111-11EZ####, Step 50: s3, #1111-11EZ####

Step 51: s4, #1111-11EZ####, Step 52: s4, #1111-11EZ####

Step 53: s5, #1111-11EZ####, Step 54: s6, #1111-11EZ####

Step 55: s6, #1111-11EZ####, Step 56: s6, #1111-11EZ####

Step 57: s6, #1111-11EZ####, Step 58: s8, #1111-11EZ####

Step 59: s8, #1111-11EZ####, Step 60: s8, #1111-11EZ####

Step 61: s8, #1111-11EZ####, Step 62: s9, #1111-11EZ####

Step 63: s1, #1111-11EZ####, Step 64: s10, #1111-11EZ####

Step 65: s10, #1111-11EZ####, Step 66: s10, #1111-11EZ####

INPUT PROCESSING FOR THE DIVISION

The subroutine concept used to implement the division operation [12]. The input given here is unary ‘1’. Ex. The number 4 represented as 1111.

Input: #1111-11####

s0, #1111-11####

Step 1: s1, #1111-11####, Step 2: s2, #1111-11####

Step 3: s2, #1111-11####, Step 4: s2, #1111-11####

Step 5: s2, #1111-11####, Step 6: s3, #1111-11####

Table. IV. Transition Table of the Turing Machine that perform the division operation

Step 10:

Step 28:

Step 36:

Step 44:

Step 52:

Step 60:

Step 68:

Step 76:

Step 84:

Step 92:

Step 100:
Step 67: $s_{10}, \ldots |Z\ldots |s_{11}$,

$h_{n}, \ldots |Z\ldots |s_{11}$,

The Turing Machine reached the halting state for acceptance. The final input string is $\ldots |Z\ldots |s_{11}$, i.e two 1’s. $4 \div 2 = 2$. Hence the construction of the Turing Machine is correct for the division operation.

IV. CONCLUSION

The Turing Machine bi-directional movement both in forward and backward direction supports the implementation of the basic calculator. This capability can be used to build the Turing Machine for finding modulus value for the given input.

REFERENCE