TMD-1 Quick Start Guide

A close up of text on a white background

Description automatically generated

Welcome to TMD-1 the state of the art in Turing teaching technology.

**What is a Turing Machine?**

The Turing machine was invented in 1936 by Alan Turing. By providing a mathematical description of a simple device capable of arbitrary computations, he was able to prove the properties of computation in general.

A Turing machine mathematically models a mechanical machine that operates on a tape. More explicitly, a Turing machine consists of:

* A tape divided into adjacent cells. Each cell contains a symbol from some finite alphabet. The alphabet contains a special blank symbol and one or more other symbols. Cells that have not been written before are assumed to be filled with the blank symbol.
* A head that can read and write symbols on the tape and move on the tape left and right one (and only one) cell at a time.
* A state register that stores the state of the Turing machine, one of many. Among these is the special start state with which the state register is initialized.
* A finite table of instructions that, given the state the machine is currently in and the symbol it is reading on the tape (symbol currently under the head), tells the machine to do the following transition steps in sequence:

1. Write a symbol from the finite alphabet replacing the one that was there. Note that the symbol written might be the same as before or different
2. Move the head either one cell to the left or one cell to the right.
3. Assume the same or a new state as prescribed by the go to state.

**TMD-1**

The Turing Machine Demonstrator (TMD-1) will have the following characteristics:

* One tape with 10 cells and a single head.
* The alphabet used will have three symbols: {0, 1, b}. 0 will be the blank symbol and b is an endmarker symbol that can be read from the tape but not written.
* There will be three states: {A, B, C}. A will be the start state plus there is a special HALT state H.

**Working With TMD-1**

To get started let's deconstruct the Tape and Finite State Machine Panels and take a look at the main components of the machine and find out what they do.

**TAPE**

|  |  |  |  |
| --- | --- | --- | --- |
| ***A picture containing clock  Description automatically generated*** | A picture containing object, clock  Description automatically generated | A picture containing indoor, sitting, clock  Description automatically generated | A picture containing clock  Description automatically generated |
| ***Tape Controls:*** |  |  |  |
| **Reset:** When pressed will set all input area cells to '0', position the Head at the rightmost input area cell, set the current STATE to 'A', and set the Next Transition Step to READ. | **Left:** Move the Tape Head one cell to the left if after a Reset. If in RUN mode will make TMD-1 run slower. Can also be used to position the Head prior to running. | **Flip:** Flip the symbol at the current Head position. In conjunction with Left and Right allows you to setup "data" in the input area. | **Right:** Move the Tape Head one cell to the right if after a Reset. If in RUN mode will make TMD-1 run faster. Can also be used to position the Head prior to running. |
| A picture containing clock, front, sitting, hanging  Description automatically generated | | | |
| ***Tape:*** Turing machines are all about the tape. While some Turing machines have tapes that are infinitely long in one or both directions, TMD-1 has a more modest fixed tape length of ten cells. The left and rightmost cells have a special fixed symbol 'b' called an endmarker which can be read but not written on. The other eight cells comprise the "input area" of the Tape and can be set to either '0' or '1'. | | | |
|  | | | |
| ***Head:*** One lamp will be ON at any given time indicating the current position of the Tape Head. | | | |

**FINITE STATE MACHINE**

|  |  |  |
| --- | --- | --- |
| **A close up of a piece of paper  Description automatically generated**  ***Transition Steps:*** The transition step to be executed **next** is highlighted. This is the step that will be executed when the PLAY button is pressed. | ***State Register:*** The current active state will be highlighted here. The state 'A' will always be the starting state.  A close up of a keyboard  Description automatically generated  ***Transition State Table:*** This is where you "program" TMD-1 by defining the transitions between states. You do this by placing tiles in the blank spaces that are appropriate to the Transition Step row they are being placed in.  A close up of a calculator  Description automatically generated  For WRITE row tiles are '0' and '1', for MOVE 'L' and 'R', and for GOTO 'A', 'B', 'C', and 'H'.    ***Transition Indicator:*** Once the READ step has been performed, the current active transition column determined by the combination of current state and symbol being read will be highlighted. | ***Finite State Machine Controls*:**  **RUN** - TMD-1 will run the program until the HALT state is reached.  A picture containing meter  Description automatically generated  **STEP** - Each time the PLAY button is pressed a Transition Step will be executed**.**  **A picture containing clock  Description automatically generated**  **PLAY -** Use the PLAY button to start the machine RUNning or to execute the Next Transition Step.  **A close up of a sign  Description automatically generated**  **HALT -** Will light up when TMD-1 has reached the HALT state. |

**On System Start or Reset**

When the machine is first turned on or has just been Reset you should see:

A close up of a device

Description automatically generated

This means that TMD-1 has been initialized with the following starting settings:

* All cells in the input area have been cleared to 0s.
* The Tape Head is set to the rightmost cell of the input area.
* The State Register is set to the default starting state of 'A'.
* The Next Transition Step to be executed has been set to READ.

In other words, TMD-1 is ready to go.

**Your First TMD-1 Program**

With the "introductions" out of the way we are going to jump into the deep end and create our first Turing Machine Demonstrator program. The problem is simple:

*Write a program to invert all of the symbols in the input area. So all 1s become 0s and vice-versa.*

Let's get started.

**Laying Some Tiles**

Programming TMD-1 is as simple and filling in the State Transition Table with the provided tiles. Let's see how this works.

When that first READ is executed, the symbol read could be any one of '1', '0', or 'b'. Since we are in the 'A' state, we will have to fill in all three Transitions associated with those symbols and 'A'.

Starting with '1', since we are inverting we are going to want to WRITE a '0' to the cell. All of the remaining input cells to be processed are to the left of the current Head position so MOVE should be an 'L'. We will stick with 'A'' for the GOTO state, since there may be more 1s to process in the same way. With the '1' Transition column done:

A close up of a device

Description automatically generated

The '0' Transition will be very similar except we want to WRITE the inverted '1' symbol:

A close up of a device

Description automatically generated

And finally, if we encounter the 'b' symbol we know that the whole input area has been processed so just HALT. It does not matter what MOVE direction we use here:

A close up of a device

Description automatically generated

So that's it. Our first program is finished, using only a single state to accomplish the task. But is the program correct? We should probably STEP through the first Transition to make sure everything is working as expected.

**Slow and Steady**

Using the STEP feature is a great way to debug a TMD-1 program.

Before we can start testing the program we just "entered" we have to press the Reset button in order to "load" the State Transition Table into TMD-1's memory. Having done that, set the RUN/STEP toggle switch to STEP and press the PLAY button once:

A close up of a device

Description automatically generated

We can see that the READ operation has selected the 'A0' Transition to execute which is correct since we are in state 'A' and the Head is on a '0' cell. The Next Transition Step has been set to WRITE. Check!

Press the PLAY button again:

A close up of a device

Description automatically generated

WRITE was executed and the Head's cell has been updated to a '1'. Next Transition Step set to MOVE. Check!

Press the PLAY button again:

A close up of a device

Description automatically generated

Next Transition Step set to GOTO. Head has been correctly moved one cell to the left. Check!

Press the PLAY button one more time:

A close up of a device

Description automatically generated

GOTO has Transitioned back to the 'A' state and is ready to READ the next cell. Check!

All is looking good. Time to RUN the program and see what happens.

**Running**

To run the program, press the Reset button again to re-establish a clean starting state and switched the RUN/STEP toggle to RUN. Use the Tape Controls to put a few 1s into the input area to get some good test data. Position the Head back at the rightmost input area cell (remember that this is the second cell from the right as the **b** cell is not part of the input area) before pressing the PLAY button.

**Your Next Program**

Hopefully, I've convinced you how easy it is to write a TMD-1 program. A good next step might be to extend the inversion program we just wrote.

If you consider the input area with its 0s and 1s to be a binary number, then what the inversion program does is to create what is called the 1s compliment of that number. The 2s compliment of a binary number is just the 1s compliment of the number to which a 1 is added (mathematically).

You have 2 whole unused states to add the 2s compliment functionality. Are you up to the challenge?