

# Keyboard Velocity Curve Measurements and Implementation

Duane Strong, John Fertig, Si Moorehead

duanes@strongenging.com

*03/13/02*

## Introduction

This document describes the method used to implement keyboard velocity curves on the Lego project. It describes the method used to characterize other keyboards such that they can be emulated as well as the implementation of the velocity curves in the Lego keyboard.

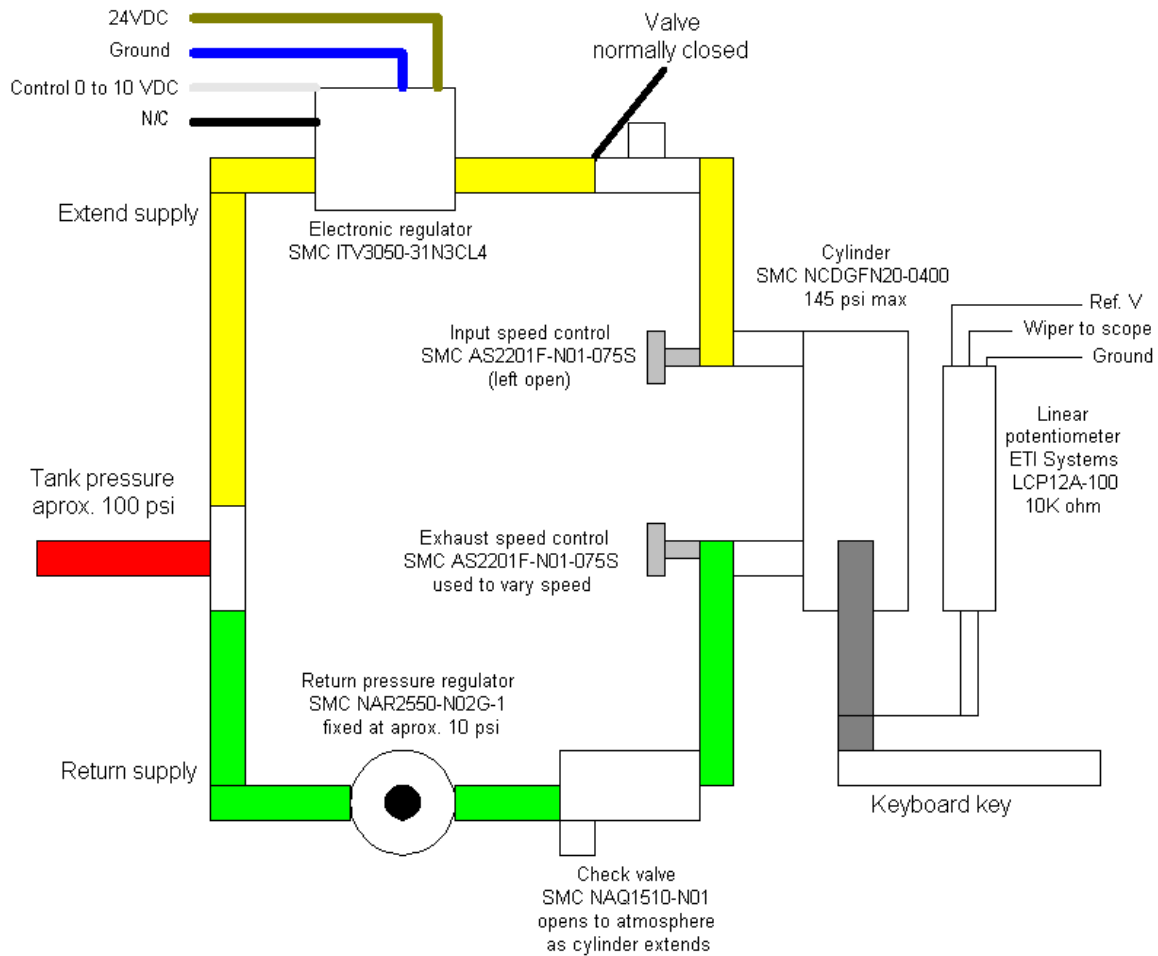
A velocity curve can be thought of as a function whose domain is a linear velocity applied to the keyboard key and whose range is a MIDI note velocity number.

## Design Goals

Rather than start from scratch it was thought that by characterizing other well considered keyboards velocity curves we would at least be much closer to converging on good curves, and at most we could capture and replicate the curves of these keyboards. A test fixture was designed to apply controlled, repeatable linear velocities to a keyboard and record the MIDI values returned. Initial experiments dropping a wooden dowel from various heights and computing the velocity at impact due to the acceleration of gravity gave the operational range that this test fixture would have to meet. A range of 2 cm/s to 125 cm/s was determined.

## Test Fixture

From the experimental data multiple ideas were considered but it was felt that pneumatics would be the best way to achieve the 125 cm/s goal. After consulting with the vendor an appropriate actuation cylinder was found.



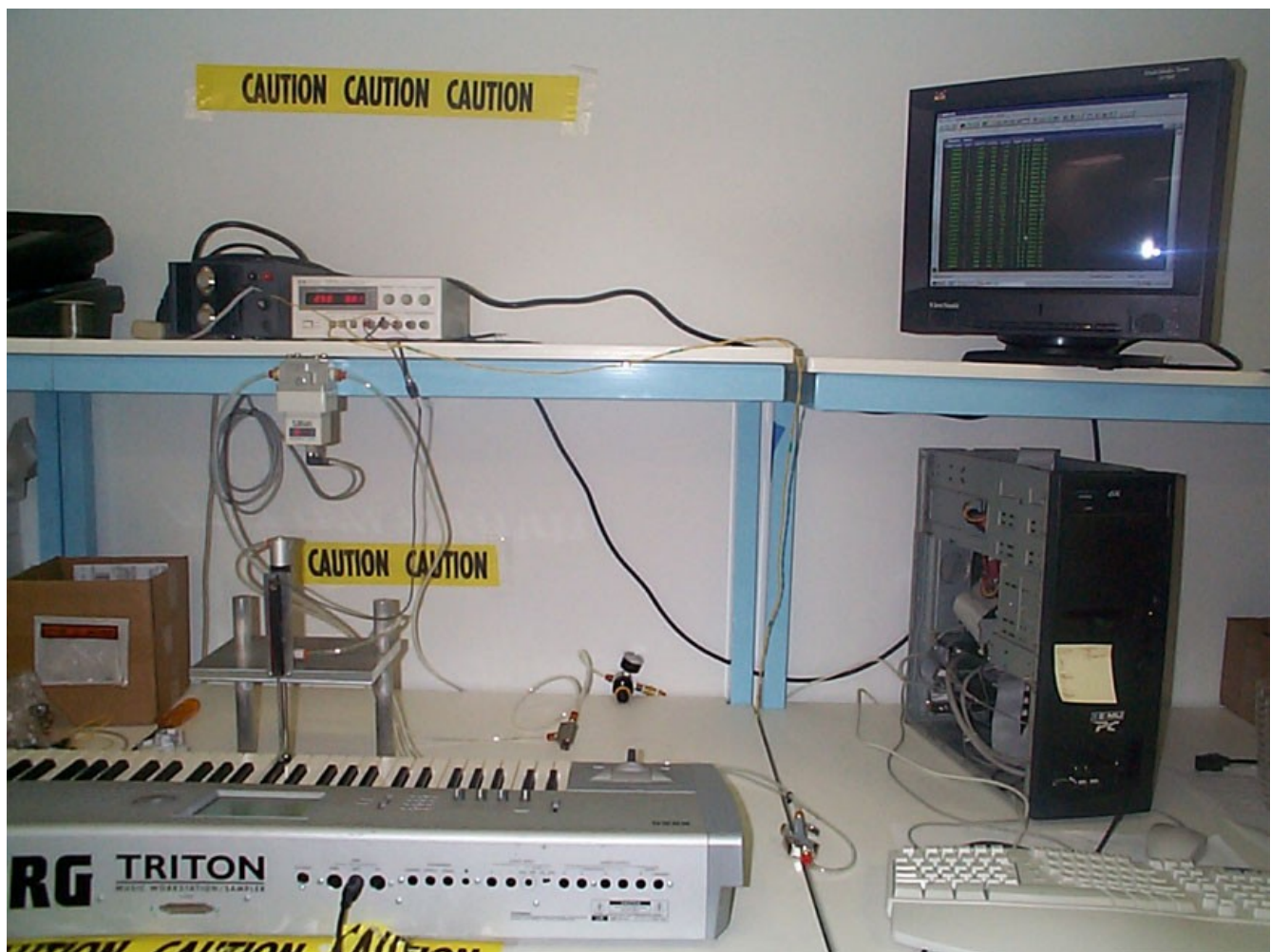
*Illustration 1: A block diagram of the test fixture*

A compressed air supply of approximately 100 psi is split into two sources by a T connector. One source is used to extend the cylinder, the other is used to retract the cylinder. The retraction pressure is fixed by a regulator to about 10 psi. The 10 psi source is then passed through a check valve that provides a path to atmosphere for the cylinder exhaust when it exceeds 10 psi. Once the exhaust pressure falls below 10 psi the check valve closes and the 10 psi returns the cylinder to its fully retracted position. The extend source pressure is set by an electronically controlled regulator. At first it was thought that this regulator could be controlled such that it would fire the cylinder at the various velocities required. The electronic regulator had two problems that made it unsuitable for this. First the regulator took a few seconds to achieve the desired pressure, and second it did not offer enough control for the slower velocities. A momentary type valve was added after the regulator such that the regulator could come up to the desired pressure, and then the cylinder could be fired by opening the valve. The fine control of the cylinder speed was achieved by leaving the regulator at a maximum pressure of 90 psi and using the mechanical speed control regulator on the exhaust port of the cylinder as it extends. By switching off the regulator control voltage (via a footswitch) the extend pressure is zero (vented to atmosphere) and the retract pressure automatically returns the cylinder rod.

A linear potentiometer is linked to the cylinder rod and provides a voltage proportional to position to the storage oscilloscope. From the scope screen a reading of cm/s can be taken at the end of the cylinder rod travel. The last 2/3 of the travel is quite linear.

The cylinder is mounted to a metal plate that slides on two bars which are bolted to the table from beneath. Set screws fasten the plate to the bars such that it can be adjusted for height to accommodate different keyboards and back vs. white keys.

The workstation consists of the pneumatic jig mounted to the table, a 24VDC supply for the regulator, a dual adjustable supply for the regulator control voltage and the potentiometer reference voltage, the keyboard under test, and a PC running midiox to record the MIDI velocity output of the keyboard.



*Illustration 2: Workstation*



*Illustration 4: View with the rod extended*

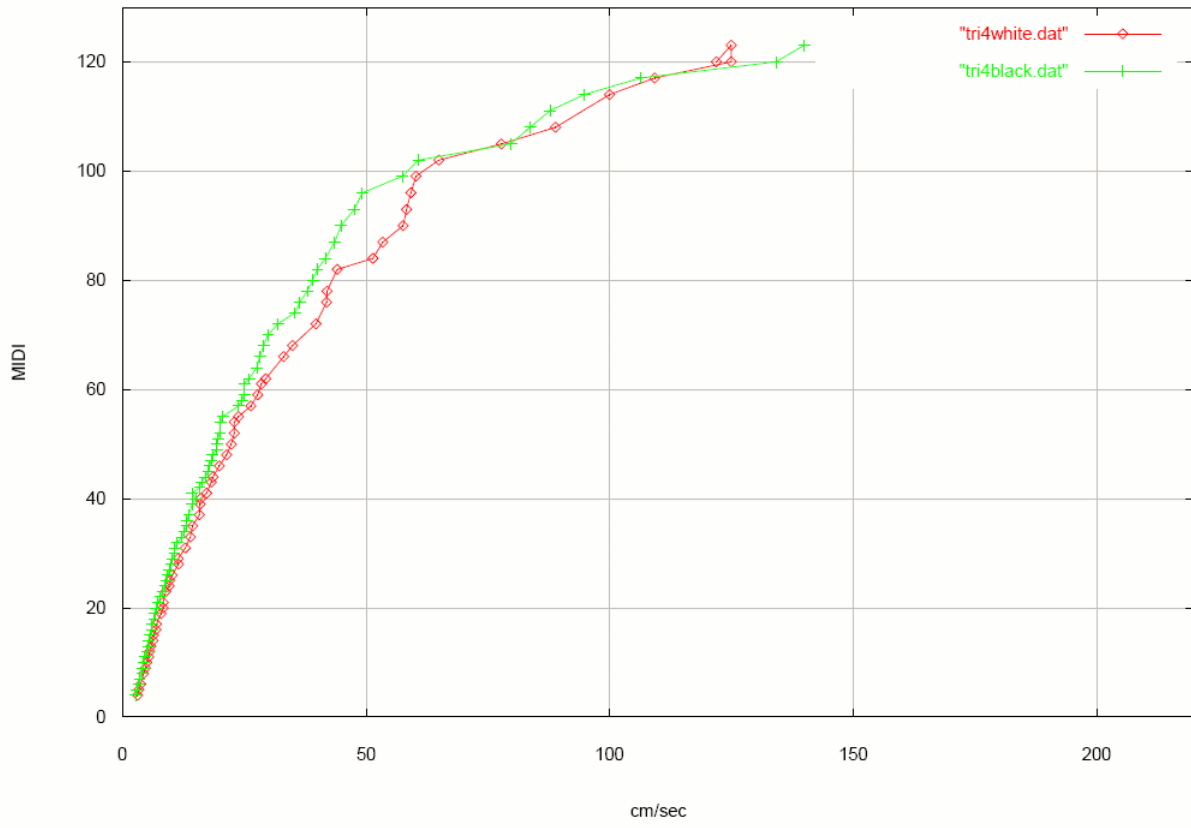


*Illustration 3: View with the rod retracted*

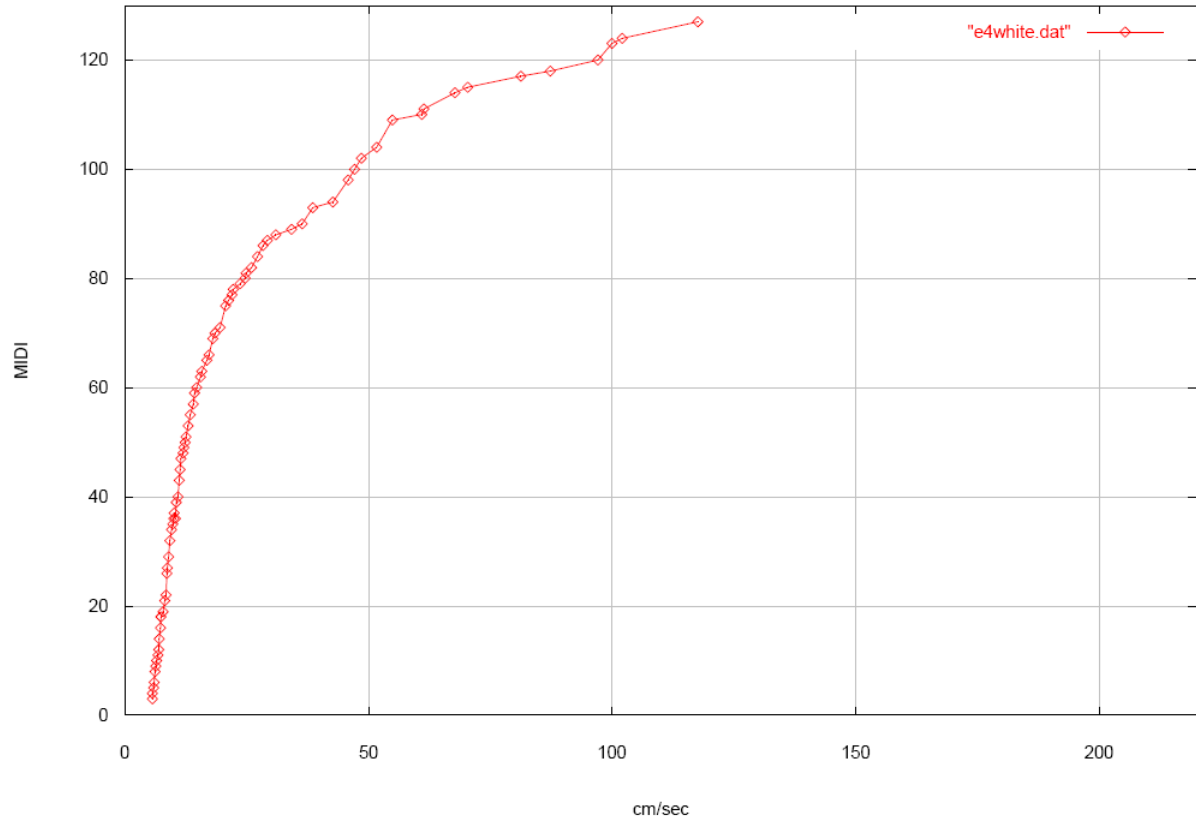
## Data

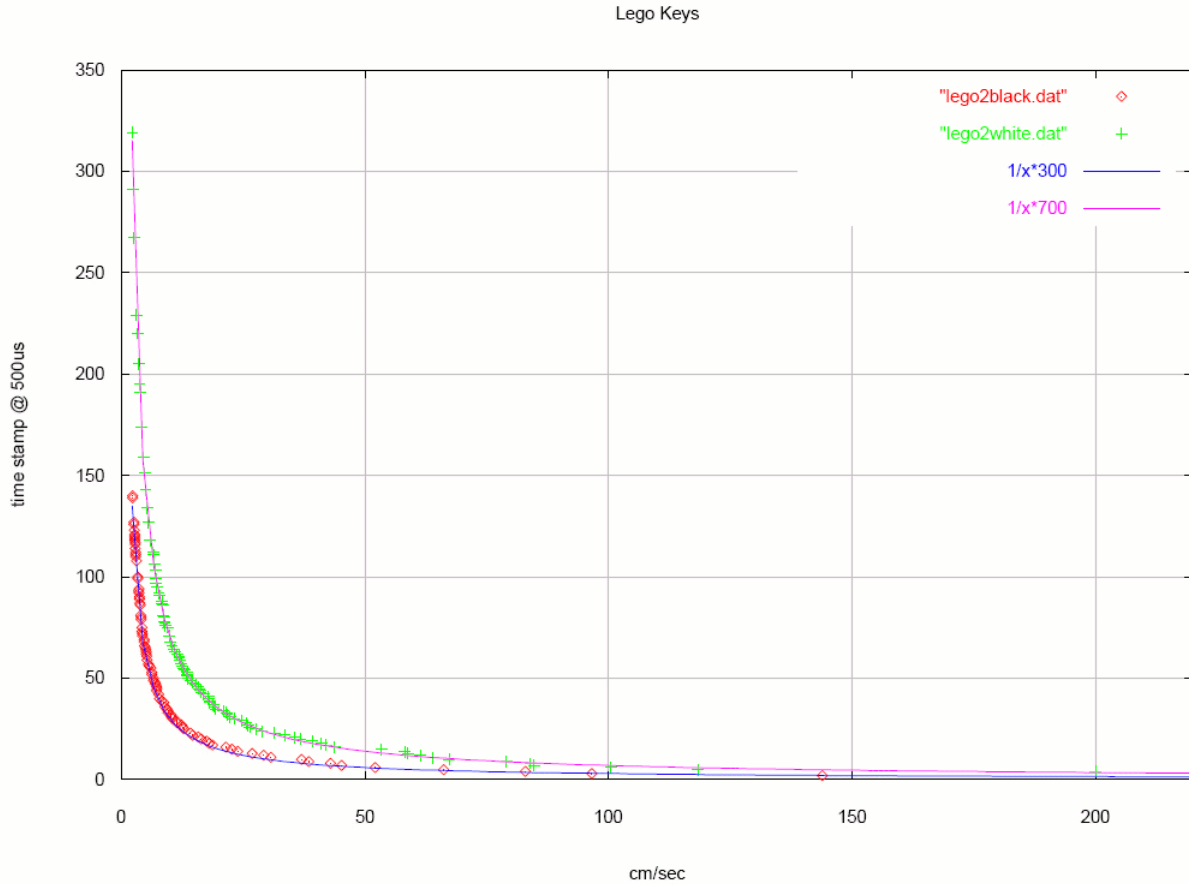
Data is obtained from a keyboard by adjusting the exhaust speed control and extending the cylinder repeatedly until the next MIDI value appears on the PC MIDI monitor (midiox). The test fixture was used to gather data on the following keyboards:

Korg Triton Table #4



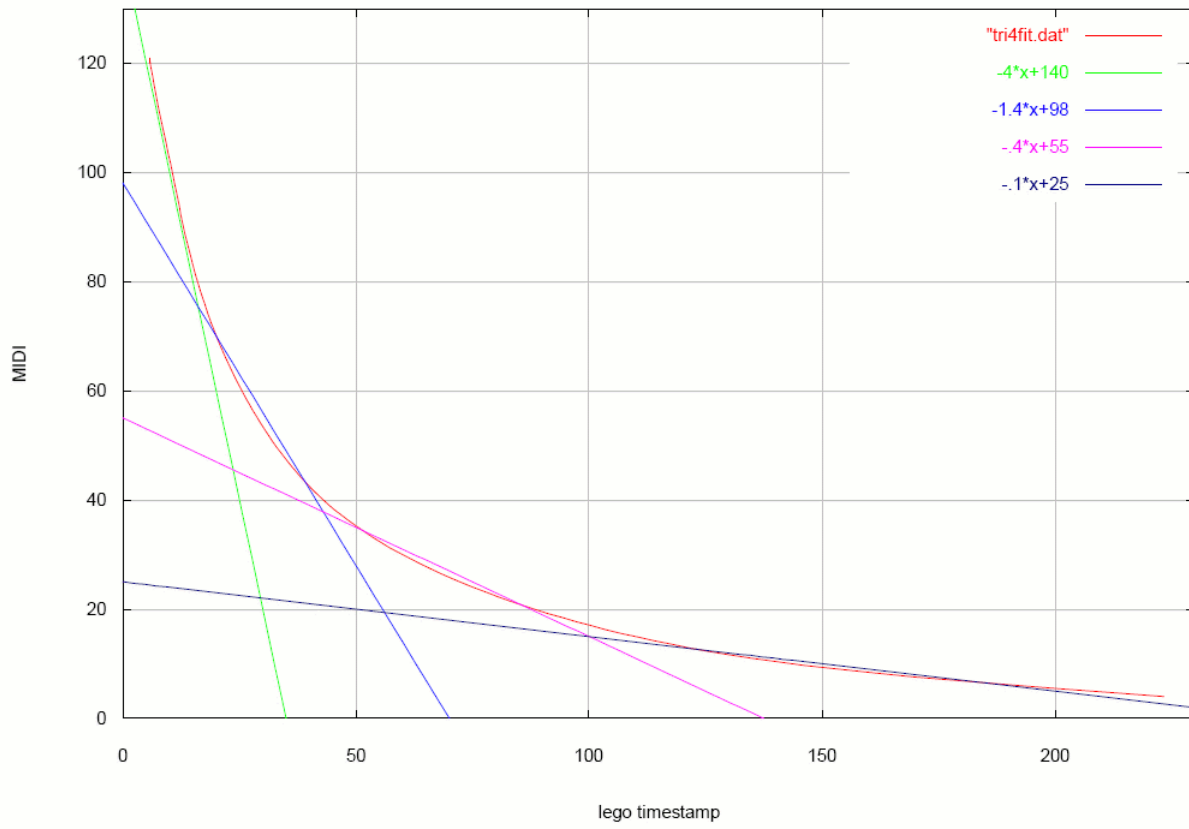
E4K



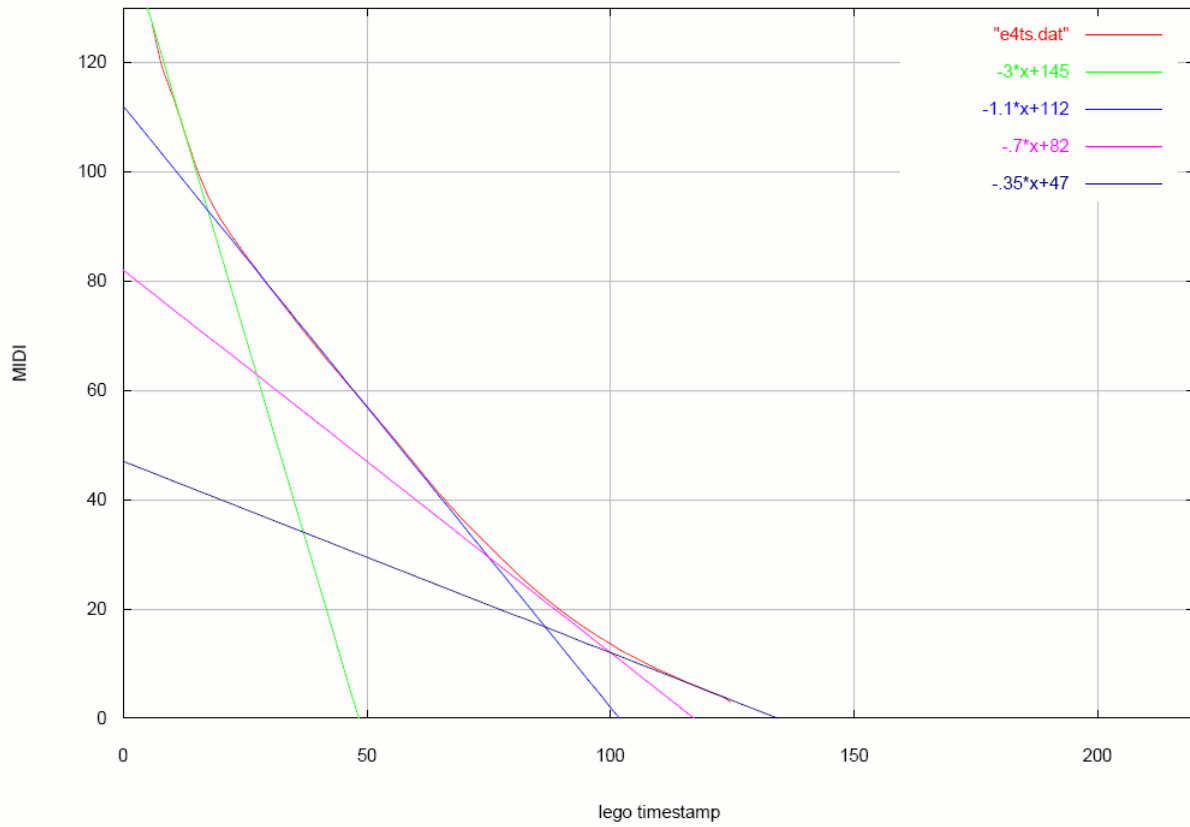


The end point velocities gathered in the Triton testing were replicated on the Lego keybed and Lego switch closure times were observed on the scope. From this data it was determined that the Lego would need to scan each key switch with a minimum resolution of 500u seconds and maintain a time stamp of at most 280 samples or 9 bits. For improved performance a sample rate of 250uS would be desirable, and a 10 bit time stamp would result. Lego currently uses a 500uS sample rate but maintains a 10 bit sample for future expansion. The Lego keybed with a 500uS scan rate was then put back into the test fixture and the velocities replicated. Special software in the Lego printed the resulting time stamp values out the debug serial port to the PC running hyperterm terminal emulation software. From this data a velocity to time stamp function was observed for white and black keys. There was no noticeable difference among (other) white keys, and no noticeable difference among (other) black keys. Curves were fit to the data and the functions  $y = 1/(x \text{ cm/s}) - 700$  for white and  $y = 1/(x \text{ cm/s}) - 300$  for black were determined. These functions were then applied to the original velocity data to arrive at timestamp to MIDI curves for Triton and E4K. These curves were then approximated with a four part piecewise linear function, which is how the velocity curve function is implemented in the Lego OS.

Triton #4



E4K



## Velocity Curves

The time stamp information from the Lego PIC keybed scanner is converted to MIDI velocity values by subjecting the time stamp to one of four linear functions. This technique was chosen for a number of factors; The 10 bit timestamp value does not lend itself to small lookup tables, four linear functions are easier to manipulate, it maps more closely to how people think about velocity, documents from Ensoniq indicate that is how they did it. A development environment was created in the Lego OS where the four segments of the function could be typed in via a debug terminal. The debug user can enter the slope and y intercept for each of the four segments. The software finds a solution for the intersection point between each successive linear function using Cramer's rule (this technique can be found in any linear algebra text). It is only necessary to perform this calculation when velocity curve segments are changed. The time stamp is compared to the X value of each solution to find which segment it belongs in. This segments linear function is applied to the time stamp yielding a MIDI velocity number.

It was an extremely useful feature to be able to give velocity curve authors quick visual feedback on the changes that they wished to make to a velocity curve. Using gnuplot on a PC to plot out the four segments of the velocity curve allowed them to see what they were doing and try out changes on the fly. After seeing what they wanted the resulting linear functions were typed into the debug terminal and they could hear the changes immediately.