Reconstructing the Unveiling Demonstration of the ENIAC

Brian Stuart [1]

[1]Drexel University

October 30, 2023

Abstract

Reports on a reimplementation of the ENIAC programming demonstrated at its unveiling on Feb 15, 1946. The reconstructed demonstration is shown in the YouTube video: https://youtu.be/SGBT2Danh-g

Article

Reconstructing the Unveiling Demonstration of the ENIAC

Brian L. Stuart
Drexel University

***Abstract*—What must it have been like to have seen the ENIAC for the first time at it's unveiling, to learn of this new type of machine, to see computations happening faster than previously imagined? Here we discuss a reconstruction of the demonstration conducted at the unveiling. We piece together available information about the event and create a configuration to run on a simulated ENIAC to create an opportunity to relive the experience of seeing it for the first time.**

On February 15, 1946, the Electronic Numerical Integrator and Computer (ENIAC) was unveiled at the University of Pennsylvania, Moore School of Electrical Engineering. Designed by John Mauchly and J. Presper Eckert, this machine had been in development from 1943 to 1945 in secret under contract for the US Army. With World War II being over and a substantial need for such a machine being foreseen, the Army decided to lift the secrecy and make the machine public. As part of the unveiling, the development team prepared and conducted a demonstration of the machine for those attending the event.

The year 2021 marks the 75th anniversary of the unveiling, and a number of virtual events were held to commemorate the occasion[1], [2]. As part of the observations of the anniversary, we set out to release a video reenactment of the demonstration. This project involved researching available records of the demonstration and reimplementing an ENIAC programming configuration that would replicate the experience. Running that configuration on the author's ENIAC simulator[3] provided the material for the video reenactment. The presentation here assumes some knowledge of the basic operation and programming of the ENIAC. The reader is encouraged to consult earlier sources for such introductions.[4], [5], [6], [7], [8]

EVOLUTION OF THE DEMONSTRATION

The demonstration performed on February 15 was not the first time that the ENIAC was demonstrated to people outside the university and military communities. On the first of the month, a demonstration was conducted for the press. This event provided news outlets with a preview of the machine and with all the information they would need to publish the first public accounts of the system. Release of the information was, however, embargoed until the official unveiling.

After the event was over, a number of the reporters asked for information on what exactly was demonstrated. In response, Arthur Burks, who conducted the demonstration, wrote a description of what computations were included[9]. This document is still available to us, because it

appeared as an exhibit in the ENIAC patent trial. We provide a transcription of it in the appendix. The stages of the demonstration described there included:

1) One accumulator loaded with 97,367.
2) 5000 additions of 97,367, giving the sum 486,835,000.
3) Repeat of the addition demonstration with the clock slowed down by a factor of 1000.
4) Loading the multiplier and multiplicand accumulators with 13,975.
5) Computation of the product 195,300,625.
6) Accumulation of 500 products, giving 97,650,312,500.
7) Generation of a tables of 100 squares and cubes with printing.
8) Recomputation of the squares and cubes without printing.
9) Computation of a table of 100 sines and cosines with printing.
10) "A problem of great importance to the Ordinance Department" but "classified as secret."

In his account of early computing, Herman Goldstine describes the last part of the demonstration as "A modification of the E-2 ENIAC run as an illustration of a long and complicated calculation." The footnote to this clarifies E-2 with the statement "The so-called E-2 run was part of the Los Alamos problem."[10] The Los Alamos problem was the earliest major problem solved on the ENIAC. It related to simulating the ignition of a thermonuclear chain reaction as part of the early theoretical work on the hydrogen bomb[11]. These simulations were carried out during late 1945 and January 1946, and further work was done on them after the unveiling.

As one might guess, a classified computation that can't be explained to the audience does not make for a very good demonstration. The original motivation for the ENIAC, artillery trajectories, would certainly be a much better example to demonstrate. To that end after this first demonstration, Goldstine and his wife Adele invited Jean Jennings (later Bartik) and Frances Elizabeth (Betty) Snyder (later Holberton) to their apartment to discuss the upcoming unveiling and demonstration. As Bartik later described the earlier demonstration, "I understand it was very boring[12]." Bartik and Holberton had been working on programming artillery trajectory computations on the ENIAC. The Goldstines asked them if the trajectory programming was ready to go and could be up and running in time for the unveiling on the 15th. In Bartik's autobiography, she describes their answer by saying, "We said we sure could."[13]

Although we have not found records of the programming details of the later demonstration, Burks did later refer to the demonstration, where he mentioned addition being demonstrated first and the trajectory being the centerpiece[14]. For purposes of this reconstruction, it is our working hypothesis that the functions demonstrated on the 15th were essentially the same as those on the 1st, but with the artillery trajectory computation replacing the Los Alamos problem. The remainder of this paper describes a reimplementation of the demonstration based on this conjecture.

SEQUENCING

For most applications of the ENIAC, a program is started and it simply runs until it is completed. The types of applications envisioned for the machine did not include any direct user interaction. However, for a demonstration, a presenter needs the ability to present the computations in phases, timed by discussion. The one mechanism on the ENIAC suitable for this purpose is the Initiating Pulse button on the Initiating Unit and replicated on the portable control station. Each time the button is pressed, a pulse is emitted on the terminal labeled Io.

For the initiating pulse to trigger different actions on each instance, it must act through the Master Programmer. The approach we adopt here is to "prime" a Master Programmer stepper as the last step in the previous part of the demonstration. Then, when the initiating pulse button is pressed, the pulse passes to the Master Programmer which in turn passes the pulse out the appropriate stepper output.

Because each stepper in the Master Programmer has only six stages but there are more than six phases of the demonstration, we use a combination of multiple steppers. Stage 1 of stepper A drives stepper C for the addition and multiplication parts of the demonstration. Stage 2 drives stepper B for the generation of the squares

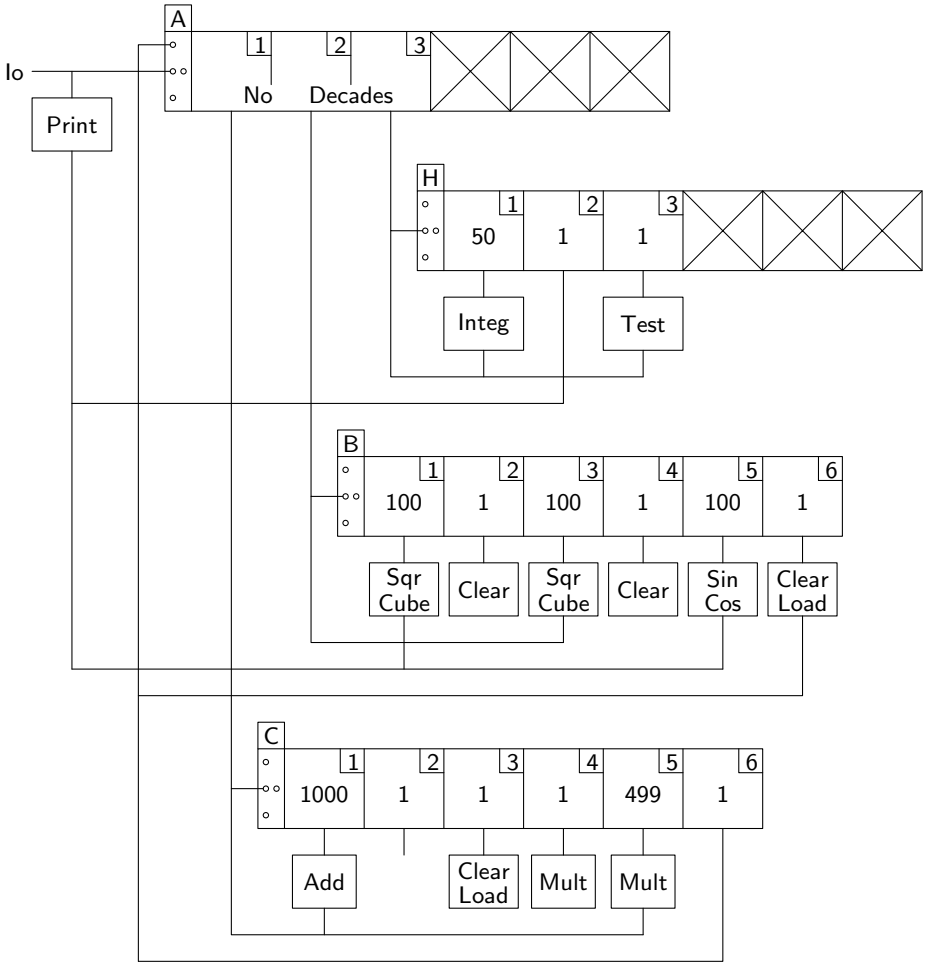

Figure 1. Demonstration Control Structure

and cubes table and of the sine and cosine table. Stage 3 initiates the simulation of the artillery trajectory. **Figure 1** shows this structure in the form of a diagram of Master Programmer links, similar to those shown in the ENIAC Technical Manual[15].

There are a few techniques that are used here that will make the control structure easier to understand. First, note that the actions on stages 2, 3, and 4 of stepper C and those on stages 2 and 4 of stepper B don't go anywhere when they are completed. This is how the demonstration pauses until the operator presses the initiating pulse button again. Second, the repetitions are typical of looping in ENIAC programming. The first iteration is triggered by a pulse on the stepper input, coming from stepper A. Then the control signal at the end of each repeated operation also drives the stepper input until the count is reached and the stepper advances to the next stage. Actions that result in a printing step operate a little differently. They drive the next iteration by sending the pulse indicating completion of the print operation to stepper A's input where the initiating pulses also go. Third, because no decades are assigned to stepper A, any number of input pulses will be routed to the same stage, until a pulse is received on the stepper direct input. The completion pulses from stage 6 of both stepper C and stepper B are routed to the stepper direct input of stepper A, shifting the operation to the next stepper. The remainder of this paper examines the details of the demonstration.

ADDITION

The first part of the demonstration centers around performing multiple additions. Prior to performing the additions, however, the descrip-

tion of the demo has a constant loaded into one of the accumulators, "The operator, with the push of a button, made the number 97367 appear in an accumulator..." Because there are several constants that are needed throughout the demonstration, we load them from a punched card at this step. Thus, we conjecture that the button referred to in this statement is the one labeled Reader Start. The initiating pulse button is then used to start the addition sequence. This is consistent with the statement in Burks' description, "By pushing another button he caused the number to be added to itself 5000 times." In the implementation here, accumulator 1 holds the constant and accumulator 2 performs the additions.

The natural way to carry out 5000 additions would be to use the Master Programmer with a stepper stage set at 5000. However, if we do this where the stepper output triggers the addition and the completion of the addition triggers the Master Programmer, we introduce a second addition time between each pair of actual additions. This would contradict the intention of showing that the machine is capable of 5000 additions per second and the report that the answer is produced in one second. The approach taken here is to set the accumulators to a repeat of five, so that for each stepper output pulse, five additions are performed. This is why stage 1 of stepper C is set to 1000 in Figure 1. After 1000 cycles of five additions, stepper C advances to stage 2 and the sequence stops.

The last part of the addition demonstration contrasts the ENIAC with its predecessors and contemporaries. In particular, the machine is demonstrated to be about 1000 times faster than other machines such as the Harvard Mark I and the Bell Labs Model V. This is described with the statement, "...the ENIAC was slowed down to one-thousandth of its normal speed..." This was most likely done by connecting a 100Hz clock to the external clock input on the Cycling Unit. In the simulator used here, there is a control on the Cycling Unit that's treated as a switch for setting the clock rate. To save stepper stages, the whole initialization sequence is restarted with the slower clock speed, including reading the punched card of constants. Because the computation at that speed would take 16 2/3 minutes to complete, it seems unlikely that they actually ran the full 5000 additions before turning the clock back up to its normal speed.

An alternative possibility is that the slower additions were performed on the two-accumulator prototype built in 1943. Analysis of the newsreel footage suggests that it was the two-accumulator prototype that had the now famous ping pong ball lights for demonstration. If indeed that was how the slower computations were demonstrated, it would have made for an especially dramatic comparison, as the entire rest of the demonstration could be conducted on the ENIAC-proper while the prototype was still performing the 5000 additions. Although it is appealing to imagine such a contrast being illustrated, the only indication that the prototype may have been involved are references to the ping pong ball lights.

MULTIPLICATION

After completing the 5000 additions twice (once at full speed and once partially at the low speed), the high-speed multiplier is the next unit demonstrated. As described, this part of the demonstration was performed in three steps, each triggered by the Initiating Pulse button. The first step loads the values to be multiplied, driven by stage 3 of stepper C. Because the multiplication is described in terms of multiplying integers, the numbers are loaded into the least significant digits of the multiplier and multiplicand accumulators (9 and 10). To minimize distraction of the audience, this stage also clears accumulators 1 and 2 of the addition demonstration. In the second step, a single multiplication is initiated by stage 4 of stepper C, and the product is shown. The third step is described as multiplying the factors 500 times and accumulating the products. Because one multiplication is already done at this point, we have implemented 499 additional multiplications in the the third step, as can be seen in stage 5 of stepper C in Figure 1. We could have performed exactly 500 multiplications in the third step by inserting a stage that cleared the product accumulators, but this would have added unnecessary complication to the sequencing logic.

Because the final sum of 500 products requires 11 digits to represent, the left-hand partial product (LHPP) and right-hand partial product (RHPP) accumulators are operated in pairs configured to allow for representing and manipulating

20-digit numbers. The final result appears in accumulators 13 and 14. (Note that the ability to work with 20-digit numbers has been added to the simulator used here since earlier reports on it[3].)

SQUARES AND CUBES

The next phase of the demonstration generates a table of the first hundred squares and cubes using the well-known relations:

$$(n+1)^2 = n^2 + 2n + 1,$$

and

$$(n+1)^3 = n^3 + 3n^2 + 3n + 1.$$

This table is generated first with a card punched for each line of the table and then again without the card punching to illustrate that almost all the time was in punching the cards. Here, we put the square in accumulator 16 and the cube in accumulator 18 so that they appear separated on the output card. Because this example is discussed at length in the technical manual [15] and in other accounts, no more detail is needed here.

SINES AND COSINES

As described in Burks' account, the sines and cosines are computed using the difference equations

$$\Delta(\sin x) = -A \sin x + B \cos x,$$

and

$$\Delta(\cos x) = -A \cos x - B \sin x,$$

where

$$A = 1 - \cos \Delta x,$$

and

$$B = \sin \Delta x.$$

The constants are given as $A = 0$ and $B = 0.000098175$ for $\Delta x = 0.1$ mil. These values correspond to the mil as an angular unit used in ballistics where there are 6400 mil in a circle.

In the implementation here, the value of B is stored in accumulator 10 with the decimal point to the left of the most significant digit. The values of $\sin x$ and $\cos x$ are stored in accumulators 18 and 20, respectively with the decimal point between the two most significant digits. When calculating the new Δ, the value is copied to accumulator 9 and the multiplication is carried out. The product is in accumulators 13 and 14 with one digit to the left of the decimal point, and 19 digits to the right. Adding five to the most significant digit of accumulator 14 rounds the result to nine decimal places in accumulator 13, placing the decimal point in the right place to add to accumulator 18 or subtract from accumulator 20 with no shifting necessary.

ARTILLERY TRAJECTORY

Unsurprisingly, the artillery trajectory is the most involved of the parts of the demonstration. At this time, we do not know of any record of the complete detailed design of the trajectory as used in the demonstration. Instead, we make an attempt to recreate the design from the ground up starting with the basic differential equations. In the 1954 Ballistics Research Lab report 889, the equations of motion are given as [16]

$$\begin{aligned}
\ddot{x} &= -E\,(\dot{x} - W_x) + \lambda_1 \dot{y}, \\
\ddot{y} &= -E\dot{y} - g + \lambda_1 \dot{x}, \\
\ddot{z} &= -E\,(\dot{z} - W_z) + \lambda_3 \dot{y} + \lambda_2 \dot{x},
\end{aligned}$$

where E is the influence of atmospheric drag, g is the acceleration due to gravity, W_x and W_z are the effects of wind, and the values of λ correspond to the effects of the Earth's rotation. We find a similar set of equations (with some changes in notation) later in the 1967 report 1371[17]. In the production of firing tables, the equations are solved numerically under "normal" conditions where there is no wind, the rotation of the Earth is ignored, and $\dot{z}$ is zero throughout. The equations then reduce to

$$\begin{aligned}
\ddot{x} &= -E\dot{x}, \\
\ddot{y} &= -E\dot{y} - g.
\end{aligned}$$

It should be noted that in these equations, we are using the same value of E in both the x and y directions. This is consistent with a simplified model used in the first half of the 20th century as described in BRL report 889. "The physical assumptions behind the present computational procedures producing firing and bombing tables are that the shell or bomb are essentially particles. For the tables discussed in this report the only concession in the equations used in the

$$
\begin{aligned}
\frac{\Delta x_0}{2} &= \dot{x}_0 \frac{\Delta t}{2}, \\
\frac{\Delta y_0}{2} &= \dot{y}_0 \frac{\Delta t}{2}, \\
x_1 &= x_0 + 2\left(\frac{\Delta x_0}{2}\right), \\
y_1 &= y_0 + 2\left(\frac{\Delta y_0}{2}\right), \\
\frac{\Delta \dot{x}_0}{2} &= -(E_0 \dot{x}_0)\frac{\Delta t}{2}, \\
\frac{\Delta \dot{y}_0}{2} &= -(E_0 \dot{y}_0 + g)\frac{\Delta t}{2}, \\
\dot{x}_1 &= \dot{x}_0 + 2\left(\frac{\Delta \dot{x}_0}{2}\right), \\
\dot{y}_1 &= \dot{y}_0 + 2\left(\frac{\Delta \dot{y}_0}{2}\right), \\
\frac{\Delta \dot{x}_1}{2} &= -(E_1 \dot{x}_1)\frac{\Delta t}{2}, \\
\frac{\Delta \dot{y}_1}{2} &= -(E_1 \dot{y}_1 + g)\frac{\Delta t}{2}, \\
\frac{\Delta x_1}{2} &= \dot{x}_1 \frac{\Delta t}{2}, \\
\frac{\Delta y_1}{2} &= \dot{y}_1 \frac{\Delta t}{2}, \\
x_2 &= x_0 + \frac{\Delta x_0}{2} + \frac{\Delta x_1}{2}, \\
y_2 &= y_0 + \frac{\Delta y_0}{2} + \frac{\Delta y_1}{2}, \\
\dot{x}_2 &= \dot{x}_0 + \frac{\Delta \dot{x}_0}{2} + \frac{\Delta \dot{x}_1}{2}, \\
\dot{y}_2 &= \dot{y}_0 + \frac{\Delta \dot{y}_0}{2} + \frac{\Delta \dot{y}_1}{2}.
\end{aligned}
$$

Figure 2. Difference Equations for Artillery Trajectory Using Heun's Method

computation to the fact that the projectile is not really a particle is that there is a drag force."[16] However, even as early as this 1954 report, there is mention of work being done to add the effect of lift as is done in modern simulations.

Both the first progress report on the ENIAC (written sometime in early 1944) [18] and BRL report 889 [16] discuss the use of Heun's method for numerically solving differential equations on the ENIAC. It seems likely that this is the approach taken in the unveiling, and it is the approach taken in this reconstruction. Applying Heun's method to the simplified differential equations results in the set of difference equations shown in **Figure 2**. As in the first progress report, we set $\Delta t = 0.02$s allowing us to perform the multiplications by $\frac{\Delta t}{2}$ by shifting two digits to the right. A simple adapter cable allows this shift to be performed much faster than using the multiplier.

Next, we turn to the evaluation of E. The Ballistics Research Lab report 1371 gives the following equation[17]

$$E = \frac{\rho v K_d}{C}.$$

Here ρ is the density of air, which we will take as a constant, though it's really dependent on altitude, y, and temperature. The magnitude of the velocity of the projectile is given by v and we use it by computing $v^2 = \dot{x}^2 + \dot{y}^2$. K_d is the drag coefficient and is found in the form of tables or graphs giving the drag for a given velocity. These data are collected experimentally for each projectile shape. Finally, C is the ballistic coefficient given by the mass divided by the square of the diameter of the projectile. We also find this as one of two expressions for E in report 889[16].

In the simulation here, the value of E is obtained by first using the high speed multiplier to square $\dot{x}$ and $\dot{y}$. The sum of the squares is shifted to obtain a two-digit number used to index one of the ENIAC function tables. The values in the table are the values of E corresponding to each value of v^2. A 105mm shell provided the characteristics for the value of C in computing the table. The values of K_d came from the drag coefficient vs. Mach number graph for shell HE 105-mm, M1 in the Handbook of Ballistic and Engineering Data for Ammunition, Volume 2[19] and reproduced here as **Figure 3**.

The final aspect of the artillery simulation to consider is terminating the simulation. We know that the simulation conducted as part of the demonstration made use of the conditional capabilities of the ENIAC to terminate the simulation when the altitude became negative. The night before the demonstration was to be given, there was still a bug in which the simulation didn't terminate but continued after the altitude went negative when Bartik and Holberton went home for the night. After thinking about it overnight,

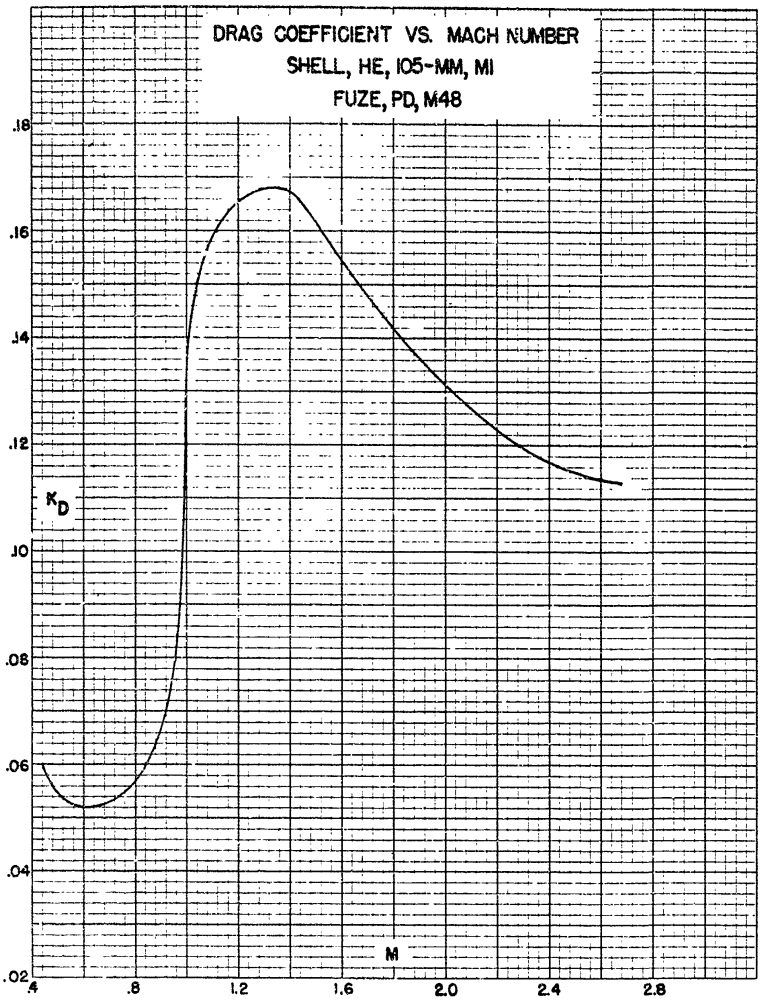

Figure 3. Drag Coefficient vs. Mach Number for 105mm Shell

Holberton returned the morning of the demonstration, changed one switch setting and fixed the bug.[12], [13]

The approach used here is that for each 50 simulation steps (each covering 0.02s for a total of 1 second of simulated time), a card is punched with the values of x and y, and the value of y is tested. If y is non-negative, then the cycle is repeated by computing another 50 time steps. The details of the testing mechanism are worth some discussion. **Figure 4** shows the subset of the set-up which implements the conditional test. Master Programmer terminal $\mathrm{H_3o}$ (stage 3 output of stepper H) is connected to control trunk line 5-10 which is also connected to terminal 12i of accumulator 20. Accumulator 20 holds the value y and its program 12 is set to transmit the contents of the accumulator subtractively. When transmitting on the S terminal, a sign of P $(+)$ is transmitted as nine pulses and a sign of M $(-)$ results in no pulses being transmitted on the PM line of the terminal. The box labeled PM in the figure represents an adapter cable that plugs into the S terminal on one end and brings the PM line of that terminal out to control trunk line 5-11. This control line is connected to accumulator 19, terminal 5i, and program 5 is configured as a dummy program as indicated by the operation switch setting of 0. On a dummy program, data is neither received nor transmitted; the accumulator transceiver is used for adjusting pulse timing. If any pulses arrive on a program input during the early part of an addition time, the program output emits a pulse at the next Central Programming

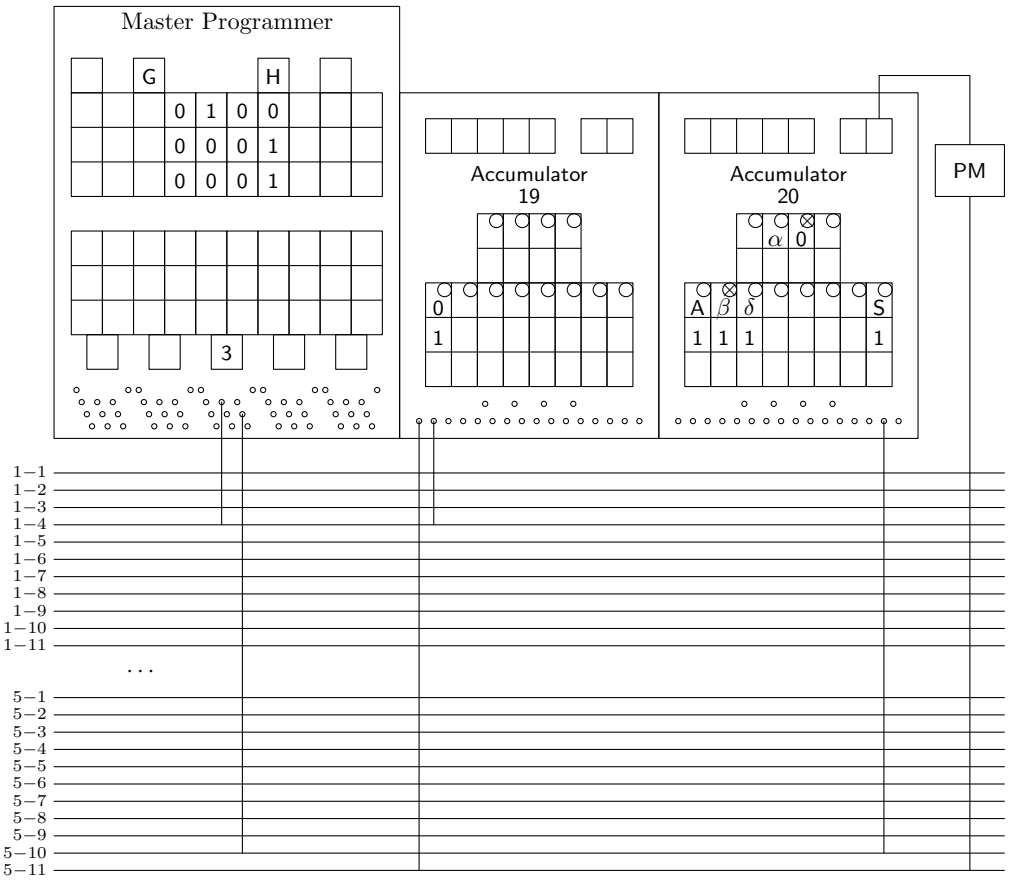

Figure 4. Configuration for Testing Altitude

Pulse (CPP) time. Of course, if no pulses arrive on a program input, then no pulses are emitted from its output. Here, the 5o terminal is connected to control line 1-4 which drives the stepper H input to start the next integration cycle. The net effect of this arrangement is that if y is non-negative after the print is performed, nine pulses will be emitted on the PM line of the S terminal, pass through control line 5-11, initiate program 5 on accumulator 19, and trigger stepper H at the next CPP time. On the other hand, if y is negative, then no pulses are transmitted on the PM line of the S terminal, program 5 is not started, and no pulse is transmitted to the H stepper input. Thus if y is non-negative, the simulation continues for another cycle, but if it is negative, then the simulation stops. **Figure 5** shows the computed trajectory for a shell leaving the gun at 1550 ft/s and a gun elevation of $22.5°$.

BEYOND THE BASIC DEMONSTRATION

Early in the development of this recreation, we were asked whether it would be able to show the bug that was fixed by Holberton on the morning of the demonstration. The reality is that we do not know enough about the details either of the programming of the demonstration or of the bug to say for sure. Bartik's account reads: "The next morning, Betty came in and knew exactly which switch on the master programmer was set incorrectly... She went over, flipped the switch over one position, and we were in business."[13] Although we can't be certain whether this is the same issue encountered the day before the unveiling, there is a master programmer switch that, when mispositioned, will result in the simulation not stopping in this reconstruction. In particular, if the stepper clear switch for stepper H were set to 2, rather than 3, then the stepper would return to stage 1 immediately after initiating the card

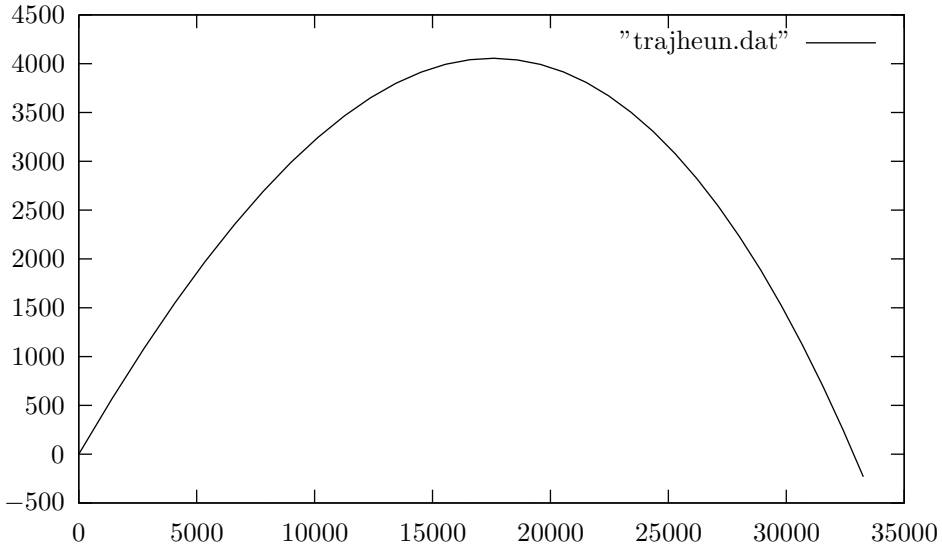

Figure 5. Simulated Projectile Trajectory

punch without performing a test on the altitude. As a result, the simulation would run indefinitely, despite the simulated shell having hit the ground.

We also know that the trajectory computation was repeated multiple times throughout the day. It could well be that the original demonstration was programmed so that after the trajectory ran to completion, further initiating pulses restarted the computation, assuming that this was planned for ahead of time. In some ways it's more interesting to speculate on how it might have been done "on the fly" during the demonstration. In this reconstruction, this can be accomplished by moving a couple of the patch cables so that the initiating pulse drives the same clear and load operation triggered by stepper B, stage 6 and the result of the clear and load drives the stepper H input. This involves unplugging the patch wire from the Io terminal to control line 1-1 and plugging it into control line 3-10. The second adjustment is unplugging the patch wire going from terminal 14o of the constant transmitter to control line 1-11 and plugging it into control line 1-4. On the ENIAC simulator, these changes can be made with the commands:

```
p i.Io 3-10
p c.14o 1-4
```

Making these changes after the full demonstration has run allows the initial pulse button to restart the trajectory simulation as many times as desired.

CONCLUSION

Seeing the ENIAC for the first time performing this variety of computations and operating at unprecedented speed must have seemed fantastic to those gathered to witness the unveiling. Reconstructing the demonstration and running it on a simulated ENIAC provides us with a way to see the ENIAC a little like those first witnesses did. We have made available on YouTube a video presentation of the reconstructed demonstration with commentary at https://youtu.be/SGBT2Danh-g. (Note that the programming run in the video is an earlier version that used Euler's method for the integration in the artillery trajectory, rather than Heun's method, described here.) Source code and related materials for the simulator used here and for the reconstructed demonstration can be found at http://cs.drexel.edu/~bls96/eniac/ and http://github.com/blstuart/eniac-simulator/.

ACKNOWLEDGMENT

The author would like to express his gratitude to Kathy Kleiman of the ENIAC Programmers Project for her assistance in getting the events surround the demonstration correct and to Bakhtier Farouk of Drexel University Mechanical Engineering Department for his consultation regarding atmospheric drag. Any remaining errors are mine alone.

Appendix

FOR RELEASE FEBRUARY 16, 1946

From Henry Herbert,
Publicity Manager,
University of Pennsylvania

At the request of several persons who attended the ENIAC press conference, February 1, 1946, the following statement concerning the ENIAC demonstration has been prepared by Dr. Arthur Burks, who conducted the demonstration.

DEMONSTRATION OF ENIAC
February 1, 1946

The operation of addition was demonstrated first. The operator, with a push of a button, made the number 97367 appear in an accumulator, where it could be read by means of the lights. By pushing another button he caused the number to be added to itself 5000 times. In one second the answer, 486,835,000 appeared in the accumulator next to the one showing 97367.

You may wonder why such a simple operation as addition was demonstrated. The answer to that is that the numerical solution of even the most complicated partial differential equation can be obtained by means of sequences of the simple operations of addition, subtraction, multiplication, and division. The important fact about the ENIAC is the speed with which it does these operations. It is 1000 times as fast as any other general purpose digital computor. To make clear how much difference this makes, the ENIAC was slowed down to one-thousandth of its normal speed and told to perform the 5000 additions. The fastest general purpose digital computor in existence before the ENIAC was completed takes 16 2/3 minutes to do what the ENIAC can finish in one second.

The high speed multiplier was demonstrated next. The number 13975 was put into the two accumulators to the left of the high speed multiplier. When the operator pushed a button these two numbers were multiplied together and the answer, 195,300,625 appeared in an accumulator to the right of the high speed multiplier. With another push of the button the operator caused 13975 to be multiplied by 13975 five hundred times and the products to be added together. The answer—97,650,312,500—appeared after 1 second.

The ENIAC next produced a table of the squares and cubes of the numbers from 1 to 100. Each square was generated from the previous number (x) and its square (x^2) by means of the formula,

$$(x+1)^2 = x^2 + 2x + 1,$$

and each cube was generated from the previous number (x), its square (x^2), and its cube (x^3) by means of the formula,

$$(x+1)^3 = x^3 + 3x^2 + 3x + 1.$$

When the square and cube of each number was computed the ENIAC stopped and the answer was punched on a card. In this manner 100 cards were punched, each containing a number, its square, and its cube. (These cards were later put through a machine which printed the table of squares and cubes on paper—see enclosed sample.)

This problem required one minute, but during most of this time the ENIAC was lying idle while the answers were being punched on cards. To show how fast the ENIAC computed the squares and cubes from 1 to 100 the problem was repeated without taking the time required for punching. The problem was finished in 1/10 second—so fast, in fact, that some who blinked didn't see it.

The ENIAC next produced a table of cosines and sines. It did this by solving the difference equations

$$\Delta(\sin x) = -A \sin x + B \cos x$$

$$\Delta(\cos x) = -A \cos x - B \sin x,$$

where A and B depend upon the intervals of the argument and are given by

$$A = 1 - \cos \Delta x$$

$$B = \sin \Delta x.$$

In the problem solved the sines and cosines were calculated in intervals of 1/10 mil, so that $A = 0$ to nine significant figures and $B = .000098175$. The sines and cosines for one hundred different angles were computed, punched on cards, and later printed in tabular form. (See enclosed sample.)

The operations and problems demonstrated up to this point were very simple and did not use much of the equipment on the ENIAC. The

ENIAC was constructed for the purpose of solving more complicated problems. The last problem demonstrated was of this character. It is a problem of great importance to the Ordnance Department, but since it is classified as secret not very much can be said about it. It involves the solution of partial differential equations. In solving it the ENIAC did in 15 seconds what a mathematician would require several weeks to do.

REFERENCES

1. "University of Pennsylvania ENIAC day: 75th anniversary of ENIAC minisymposium," 2021. [Online]. Available: https://events.seas.upenn.edu/event/eniacday/
2. "UNISYS corporation ENIAC day: 75th anniversary of ENIAC broadcast," 2021. [Online]. Available: https://vimeo.com/510236526
3. B. L. Stuart, "Simulating the ENIAC," *Proceedings of the IEEE*, vol. 106, no. 4, pp. 761–772, April 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8326772/
4. M. Bullynck and L. D. Mol, "Setting-up early computer programs: D. H. Lehmer's ENIAC computation," *Arch. Math. Log.*, vol. 49, no. 2, pp. 123–146, 2010. [Online]. Available: https://doi.org/10.1007/s00153-009-0169-8
5. L. D. Mol and M. Bullynck, "A week-end off: The first extensive number-theoretical computation on the ENIAC," in *Logic and Theory of Algorithms, 4th Conference on Computability in Europe, CiE 2008, Athens, Greece, June 15-20, 2008, Proceedings*, 2008, pp. 158–167. [Online]. Available: https://doi.org/10.1007/978-3-540-69407-6_19
6. A. K. Goldstine, "Report on the ENIAC: Part I technical description of the ENIAC," Republished Periscope File, LLC, 2012, Moore School, University of Pennsylvania, Tech. Rep., 1946. [Online]. Available: http://archive.org/details/ReportonENIACEl00MoorB
7. T. Haigh, M. Priestley, and C. Rope, *ENIAC in Action: Making and Remaking the Modern Computer*. The MIT Press, 2016.
8. B. L. Stuart, "Programming the ENIAC," *Proceedings of the IEEE*, vol. 106, no. 9, pp. 1760–1770, September 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8467000/
9. A. W. Burks, "Demonstratin of ENIAC," Memorandum, ENIAC Patent Trial, Plaintiff Exhibit 4024.5, University of Pennsylvania, February 1946.
10. H. H. Goldstine, *The Computer from Pascal to von Neumann*. Princeton University Press, 1972.
11. W. Aspray, "An interview with Nicolas C. Metropolis," Charles Babbage Institute, Los Alamos, NM, 1987. [Online]. Available: https://hdl.handle.net/11299/107493
12. J. J. Bartik, "Jean Jennings Bartik—ENIAC pioneer," Interview with Linda O'Bryon, Computer History Museum, October 2008. [Online]. Available: https://www.youtube.com/watch?v=buAYHonF968
13. ——, *Pioneer Programmer: Jean Jennigs Bartik and the Computer that Changed the World*, J. T. Rickman and K. D. Todd, Eds. Truman State University Press, 2013.
14. A. W. Burks, "Who invented the general-purpose electronic computer?" Lecture delivered in Rackham lecture hall, University of Michigan, April 1974. [Online]. Available: https://archive.computerhistory.org/resources/text/Knuth_Don_X4100/PDF_index/k-8-pdf/k-8-u2772-Who-Invented-Computer.pdf
15. A. K. Goldstine, *Electronic Numerical Integrator and Computer ENIAC Technical Manual*. Periscope Film, LLC, 2012.
16. S. Gorn and M. L. Juncosa, "On the computational procedures for firing and bombing tables," US Army Ballistics Research Laboratories, Aberdeen Proving Ground, Maryland, Tech. Rep. BRL R 889, January 1954. [Online]. Available: http://apps.dtic.mil/dtic/tr/fulltext/u2/027123.pdf
17. E. R. Dickinson, "The production of firing tables for cannon artillery," US Army Ballistics Research Laboratories, Aberdeen Proving Ground, Maryland, Tech. Rep. BRL R 1371, November 1967. [Online]. Available: http://apps.dtic.mil/dtic/tr/fulltext/u2/826735.pdf
18. "The ENIAC volume I: A report covering work until December 31, 1943," submitted in accordance with Contract #W-670-ORD-4926, University of Pennsylvania Moore School of Electrial Engineering, 1944.
19. "Handbook of ballistic and engineering data for ammunition, volume II," US Army Ballistic Research Laboratories, Aberdeen Proving Ground, Maryland, Tech. Rep., July 1950. [Online]. Available: https://apps.dtic.mil/dtic/tr/fulltext/u2/a955369.pdf

Brian L. Stuart is currently an associate teaching professor at Drexel University, Philadelphia, PA, USA. He received the B.S. degree from the Rose-Hulman Institute of Technology, 1984, the M.S. degree from Notre Dame, 1987, and the Ph.D. degree from Purdue University, 1992. Contact him at bls96@drexel.edu.