Cray XMP48 and the Cray-2 and supporting the Bellcore parallel machine. We're very happy to have Harry Reed with us today.

Harry Reed:

I sort of thought the title for my talk should be something like "My Life with the ENIAC: A Worm's Eye View." I came to BRL in August of 1950, and I guess the first thing I learned was about bureaucracy. I showed up at Personnel, and they informed me that, "By golly, we don't have any form so-and-so available." They said, "Not to worry. Come back after lunch. Go and visit the Ordnance Museum. Everything is all right. You're being paid, so you don't have to worry about it." By the end of the day, I finally met Barkley, and as he pointed out, he said, "Welcome aboard," and I was a little taken aback, because I thought this was an Army installation, but apparently it wasn't.

I wanted to say a few words about how computing was at that time and how, to a certain extent, that would shape the nature of the ENIAC. Indeed, some of the traditions that came about during this period we tended to live with quite a while.

To understand ENIAC, of course, for those of you who don't know too much about it, think about it as a processor chip with 20 register positions. That was it. That was the RAM that you had available to you, and about half of those register positions were involved intimately in the various arithmetic operations and not available for general storage. You had the function tables, which contained 3000 decade (10-position) switches, that could be set ahead of time (like a read-only memory). And of course, you had cards for input and output, which were used for intermediate storage.

The ENIAC itself, strangely, was a very personal computer. Now we think of a personal computer as one which you carry around with you. The ENIAC was actually one that you kind of lived inside. And as Barkley pointed out, you could wander around inside it and watch the program being executed on lights, and you could see where the bullet was going—you could see if it happened to go below ground or went off into space or the wrong direction. So instead of your holding a computer, this computer held you.

Given its somewhat fragile nature, there was a sort of intimate contact with it. Probably one of the biggest problems we had was the IBM cards. The only rooms at BRL that were air-conditioned were those that were used for the handling and storage of IBM cards. Nothing else got air-conditioning, but those rooms did, because the IBM cards had a nasty habit of soaking up moisture, and the readers and printers that we had in those days were extremely intolerant of changes in the size of the cards.

Interestingly enough, about 50 years later, I find that some of our computer rooms have more problems associated with air-conditioning than they do with things associated with electronics.

Given the limited storage—extremely limited storage—on the ENIAC, things like the use of subroutines were just out of the question. There was no way that you could have a prewritten subroutine—that you could plop in—because you had nothing in the way of free space that you could use for local variables and/or stacks and/or things like that. So you might have a routine that you would embed in your program, making sure that it didn't interfere with other storage. In fact, you would usually replicate the subroutine, because you didn't have enough storage to put a return address to get back to where you were coming from. So it was easier just to put in the subroutine and then not have to worry about transferring back.

To give some idea of how crowded this thing could become, Barkley at one point asked me to program a guided missile for the General Electric Company. I said, "There is not enough room; there are too many variables." And he said "Oh, go do it anyway." So, it ended up we had to split the registers and store two things in each one of those, which added to our agony. We had to use all 3000 switches on the function tables, which at that time were being used for both program storage and for data storage. And just to make everything fit, we still had to write a special piece of *microcode*, if you want to call it that; we had to rewire a new instruction because we had to preserve one register and some of the transfer operations. So this was the kind of thing that you had to do, to get a program on the ENIAC.

Setting the programs up was something like a several-week job, writing the codes and figuring out how you were going to do it. Then you spent a couple of hours turning switches—depending on how many people you could draft to help do that process. And then you spent a certain amount of time in trying to figure out whether what you had done worked or not.

One of the few cases in which you actually fooled around with the wiring of the ENIAC was when you would pull the plug off the cable that sent the command to trigger the next instruction and then would walk around the computer with a little box and a button, and push the button and watch as the various numbers bounced around from place to place as you had programmed it.

ENIAC operation itself was done with a mathematician and an engineer on each of the 8-hour shifts, and you ran 24 hours. So you had a mathematician who was responsible for getting the program together, and who would go out and recruit two other mathematicians to supervise on the other shifts, and then you would run for 24 hours a day. It was a darn nuisance to set the program up too many times.

I might also mention that, as far as reliability was concerned, the ENIAC was rather remarkable. You hear lots of numbers about failure rates and so forth, but once you got the computer settled down, the computer had a habit of running for about a couple of weeks with no errors whatsoever. This is quite a remarkable achievement.

The whole thing had a certain amount of *bailing wire* in it. People talk about the "bailing wire days of aviation." Barkley was lucky I guess, or maybe he just tells the story that way. When he was with the President, he was able to show him a trajectory. Every year at springtime, the West Point graduating class would come down to the Proving Ground. They would visit all the various functions and get a demonstration of big guns firing. Among the things they came to see was the ENIAC. So you had this troop of cadets wandering through the room.

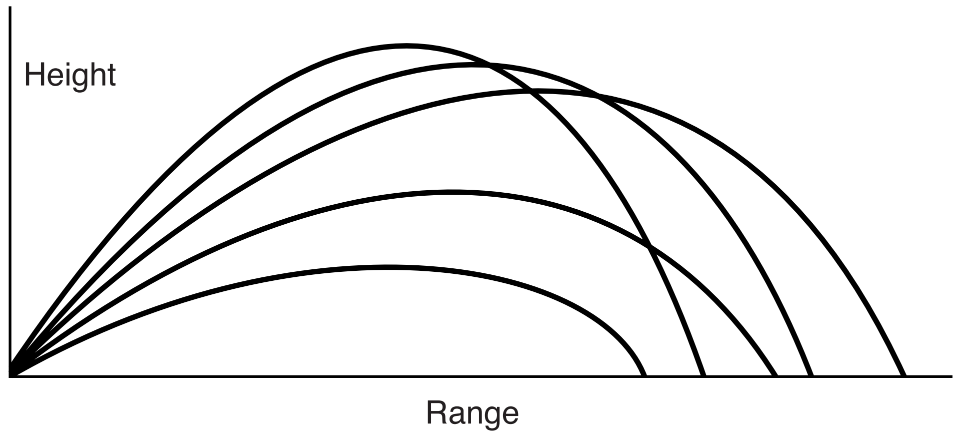
And when you had large groups of people wandering through the ENIAC room, things always went wrong. People always bumped cables and so forth and so on. So we would usually take out a deck of punched cards that contained some special diagnostic tests, and we would load these cards into the ENIAC and run those, because we won't have to worry about whether or not our results are any good. It was a great display, because these tests were constructed so that you could watch the numbers sort of flow through the registers in their patterns. So it reminded you of Times Square in New York, and you could diagnose what was going on in the computer.

So we would put these diagnostic tests on, and as I said, it sort of looked like Times Square. Then this escort officer would come in with these cadets, and he'd been briefed ahead of time. He would say, "Over there, you can see in this register that this is the velocity of a bullet, and you can see how it is moving …" None of this was true. It was just these tests going on. But we got away with it, and it did look good.

I wanted to spend the rest of my time talking about the big program for the ENIAC, which was firing table calculation. The ENIAC's basic construction, the number of registers and so forth, was quite well tailored to the problems associated with firing table calculations and in particular the calculation of trajectories.

Now, the basic problem in a firing table calculation is to calculate where the bullet is going. I've got this very simple-minded trajectory here. [see figure below] These are the kinds of things the ENIAC would calculate, for all sorts of angles, elevations, velocities. Then it would generate data about where the bullet was in space and, in particular, where it would hit the ground—which, of course, was the principal concern to most of the artillery officers at that time.

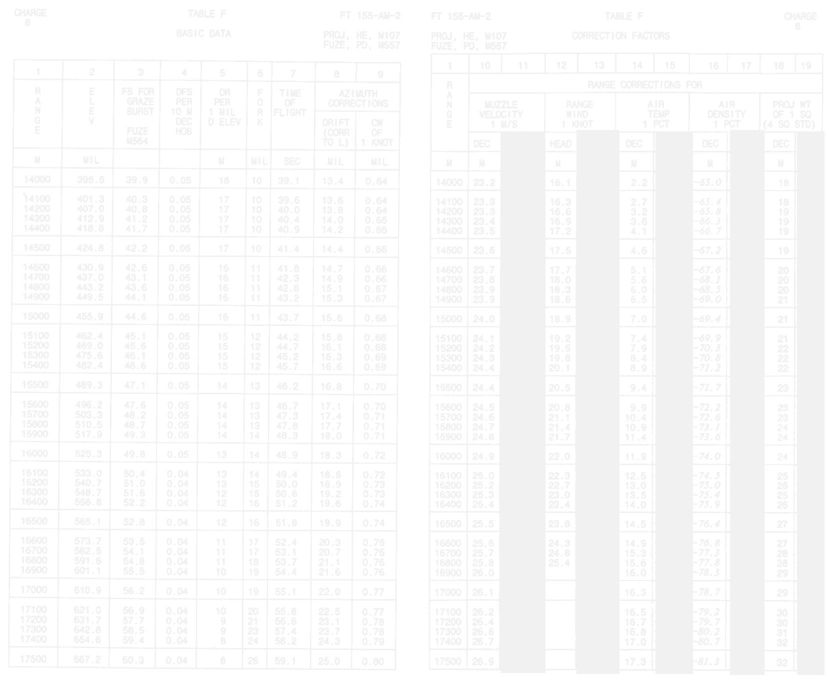
Of course, what you are getting is the wrong variables. You are getting the data for where the shell lands, given where you aimed the gun. What you really want is how to aim the gun so it will hit a specific target. That seems like an extremely simple problem for a computer today. If I worked on my PC, I obviously would have it record the data and go back through the interpolations, resort the data, and come out with what I want. Unfortunately though, it wasn't quite that simple with the ENIAC. You had no intermediate storage, so you had to take what you got out on IBM cards, and then if you were going to do something with these, you would have to resort them with a mechanical sorter, and then reprogram something on the switches so you could go back and get these new answers.



A simple set of trajectories.

Now, that actually was not all that hard to do. In fact, at one point Margery Fields from the Bombing Tables Branch was looking for some of this work, and Barkley shuffled me off on it. They wanted to do essentially the same thing for bombing tables. They wanted to generate the data for the people who designed the bomb sights rather than trajectory data. We went through quite an elaborate process of creating data and sorting it. (I thought I could qualify as a faro dealer in Las Vegas by the time I got done with the card shuffling that had to be done with this.) Essentially, we would take the punched cards, sort them, put them back, run them through rather elaborate fifth-order interpolations and smoothing processes, and generate the data.

The firing tables people were a bit more recalcitrant about that. This is a pair of pages from a firing table. This is a firing table for a 105-mm howitzer. It is just loaded with pages like that. And that's the kind of stuff we were generating.



CHARGE 8 — TABLE F — FT 155-AM-2

BASIC DATA — PROJ, HE, M107 FUZE, PD, M557

1	2	3	4	5	6	7	8	9
RANGE	ELEV	FS FOR GRAZE BURST FUZE M564	DFS PER 10 M DEC HOB	DR PER 1 MIL D ELEV	FORK	TIME OF FLIGHT	AZIMUTH CORRECTIONS	
							DRIFT (CORR TO L)	CW OF 1 KNOT
M	MIL			M	MIL	SEC	MIL	MIL
14000	395.5	39.9	0.05	18	10	39.1	13.4	0.64
14100	401.3	40.3	0.05	17	10	39.6	13.6	0.64
14200	407.0	40.8	0.05	17	10	40.0	13.8	0.64
14300	412.9	41.2	0.05	17	10	40.4	14.0	0.65
14400	418.8	41.7	0.05	17	10	40.9	14.2	0.65
14500	424.8	42.2	0.05	17	10	41.4	14.4	0.66
14600	430.9	42.6	0.05	16	11	41.8	14.7	0.66
14700	437.0	43.1	0.05	16	11	42.3	14.9	0.66
14800	443.2	43.6	0.05	16	11	42.8	15.1	0.67
14900	449.5	44.1	0.05	16	11	43.2	15.3	0.67
15000	455.9	44.6	0.05	16	11	43.7	15.6	0.68
15100	462.4	45.1	0.05	15	12	44.2	15.8	0.68
15200	469.0	45.6	0.05	15	12	44.7	16.1	0.68
15300	475.6	46.1	0.05	15	12	45.2	16.3	0.69
15400	482.4	46.6	0.05	15	12	45.7	16.6	0.69
15500	489.3	47.1	0.05	14	13	46.2	16.8	0.70
15600	496.2	47.6	0.05	14	13	46.7	17.1	0.70
15700	503.3	48.2	0.05	14	13	47.3	17.4	0.71
15800	510.5	48.7	0.05	14	13	47.8	17.7	0.71
15900	517.9	49.3	0.05	14	14	48.3	18.0	0.71
16000	525.3	49.8	0.05	13	14	48.9	18.3	0.72
16100	533.0	50.4	0.04	13	14	49.4	18.6	0.72
16200	540.7	51.0	0.04	13	15	50.0	18.9	0.73
16300	548.7	51.6	0.04	12	15	50.6	19.2	0.73
16400	556.8	52.2	0.04	12	16	51.2	19.6	0.74
16500	565.1	52.8	0.04	12	16	51.8	19.9	0.74
16600	573.7	53.5	0.04	11	17	52.4	20.3	0.75
16700	582.5	54.1	0.04	11	17	53.1	20.7	0.75
16800	591.6	54.8	0.04	11	18	53.7	21.1	0.76
16900	601.1	55.5	0.04	10	19	54.4	21.6	0.76
17000	610.9	56.2	0.04	10	19	55.1	22.0	0.77
17100	621.0	56.9	0.04	10	20	55.8	22.5	0.77
17200	631.7	57.7	0.04	9	21	56.6	23.1	0.78
17300	642.8	58.5	0.04	9	23	57.4	23.7	0.78
17400	654.6	59.4	0.04	8	24	58.2	24.3	0.79
17500	667.2	60.3	0.04	8	26	59.1	25.0	0.80

FT 155-AM-2 — TABLE F — CHARGE 8

PROJ, HE, M107 FUZE, PD, M557 — CORRECTION FACTORS

1	10	11	12	13	14	15	16	17	18	19
RANGE	RANGE CORRECTIONS FOR									
	MUZZLE VELOCITY 1 M/S		RANGE WIND 1 KNOT		AIR TEMP 1 PCT		AIR DENSITY 1 PCT		PROJ WT OF 1 SQ (4 SQ STD)	
	DEC		HEAD		DEC		DEC		DEC	
M	M		M		M		M		M	
14000	23.2		16.1		2.2		-65.0		18	
14100	23.3		16.3		2.7		-65.4		18	
14200	23.3		16.6		3.2		-65.8		19	
14300	23.4		16.9		3.6		-66.3		19	
14400	23.5		17.2		4.1		-66.7		19	
14500	23.6		17.5		4.6		-67.2		19	
14600	23.7		17.7		5.1		-67.6		20	
14700	23.8		18.0		5.6		-68.1		20	
14800	23.9		18.3		6.0		-68.5		20	
14900	23.9		18.6		6.5		-69.0		21	
15000	24.0		18.9		7.0		-69.4		21	
15100	24.1		19.2		7.4		-69.9		21	
15200	24.2		19.5		7.9		-70.3		22	
15300	24.3		19.8		8.4		-70.8		22	
15400	24.4		20.1		8.9		-71.2		22	
15500	24.4		20.5		9.4		-71.7		23	
15600	24.5		20.8		9.9		-72.2		23	
15700	24.6		21.1		10.4		-72.6		23	
15800	24.7		21.4		10.9		-73.1		24	
15900	24.8		21.7		11.4		-73.6		24	
16000	24.9		22.0		11.9		-74.0		24	
16100	25.0		22.3		12.5		-74.5		25	
16200	25.2		22.7		13.0		-75.0		25	
16300	25.3		23.0		13.5		-75.4		25	
16400	25.4		23.4		14.0		-75.9		26	
16500	25.5		23.8		14.5		-76.4		27	
16600	25.6		24.3		14.9		-76.8		27	
16700	25.7		24.8		15.3		-77.3		28	
16800	25.8		25.4		15.6		-77.8		28	
16900	26.0				16.0		-78.3		29	
17000	26.1				16.3		-78.7		29	
17100	26.2				16.5		-79.2		30	
17200	26.4				16.7		-79.7		30	
17300	26.6				16.8		-80.2		31	
17400	26.7				17.0		-80.7		32	
17500	26.9				17.3		-81.3		32	

Firing table for 105-mm howitzer.

Now people don't use these anymore. The data goes into computers that are embedded in the system, but they still print these things, because I think they ought to have them as a backup, or as a piece of history, I guess.

This contains a whole bunch of stuff starting with the *range*, the *elevation* you would shoot at, and a whole bunch of things that would tell you how you have to change things if the *wind* were blowing or if there were more *dense air* or whatever. The firing table people would take these data, and they would then perform an operation that they called "smoothing." Then they would start "differencing" all this, and create large sheets that they would write all this stuff on and then give it to a typist who would type all this stuff—and in fact, her grade was based on the fact that she went through this agony of typing these firing tables.

And I said, "Hey, I can do all that on the computer. I can even print it out for you." It wouldn't be great with an IBM tabulator, but it wouldn't be all that bad. And they said, "Oh no, no, you can't do that, because we have to smooth the data." I said, "I can smooth the data the way I did for the bombing tables people." "No," they said, "We have to do it our way." So I said, "Tell me what your way is." They said, "Well, it's hard to explain."

So I don't know whether it was sort of intellectual privacy or something they were dealing with, but they would look at me and say, "Gee, I think I better raise that number a bit or lower that number a bit." Now, nobody really cared, because anybody who thinks you can take too many of these numbers all that seriously is somewhat misled. [laughter]

But nevertheless, they did generate, among other things they called graphical firing tables, which were slide rules. And the people who made the slide rules had a terrible time if you gave them data that didn't have nice smooth higher differences. It just didn't work well when they tried to put in the graduations. But anyway, they went through this process, and it seemed terribly inefficient, although it did represent a certain trend in using the ENIAC.

Basically, the resource was extremely scarce. It was the only computer. It was busy; everybody wanted to use it—particularly after Herman and Adele[28] [Goldstine] and Clippinger[29] made it so you could set these switches and program it instead of doing all this wiring. So, what was usually done in the ENIAC was to take a high-intensity calculation problem and do that. And all the other stuff would then be relegated to other activities and to other people.

[28] Adele Goldstine, Herman's wife, was one of the ENIAC team members; Adele wrote *Report on the ENIAC (Electronic Numerical Integrator and Computer) Technical Report 1* (of 2 vols), Philadelphia, 1 June 1946. Adele later played a major role in reconfiguring the ENIAC to use the function tables for programming.

[29] Clippinger (1948).

Well anyway, that all sort of persisted in the firing table business, until we came to the 280-mm atomic cannon. Once again, I think my friend Barkley said, "Hey, we need firing tables for the 280-mm atomic cannon, and there's a problem. The problem is that with atomic things, you have to burst them up in the air." And so all of a sudden, this one-dimensional table became a two-dimensional table, because I had to know where I would aim the bullet to hit each of these points up *here* [hands over his head]. And now I had a humongous human problem.

The firing people just sort of threw up their hands. What we essentially needed was something that would look like this [see below].

Every one of these pages represented a column on this sheet. So I was talking about a lot of calculations. So finally, the firing tables people had to give up. We did the calculations on the computer. We then took the data, played with various sorting games to do the double interpolations, did all the differential effects, took all the differences, and then I put a brand new ribbon in the IBM tabulator and put some nice bond paper in there instead of the usual stuff, and actually printed the firing tables. It was quite a dramatic experience for the folks. I don't think they ever felt that a computer could take over their business.

There was a tendency, particularly as I say with the ENIAC, to reduce the problem to something that would fit on the computer. Now when the human computers were calculating trajectories, they used very high-order numerical integration techniques, probably fourth- or fifth-order integration techniques at relatively large intervals. So, it was essentially a minimal computational problem, but unfortunately it was a fairly sizable memory problem. The ENIAC couldn't handle that. It didn't have the data storage necessary to do high-order integrations; it just couldn't "remember" the immediate results. What we had to do then was to revert back to a very simple trapezoidal integration, and, of course, then run the integration interval back so that we now had an adequate approximation to the problem. There was a good bit of that which went on.

QUADRANT ELEVATION - MILS

Height of Target - Meters

Range Meters	-400	-300	-200	-100	0	100	200	300	400	500	600	700	800	900	1000
10000	318.3	329.9	341.5	353.2	364.9	376.6	388.4	400.1	412.0	423.8	435.7	447.7	459.6	471.7	483.7
10100	323.1	334.6	346.1	357.7	369.4	381.0	392.7	404.5	416.2	428.0	439.9	451.8	463.7	475.7	487.7
10200	327.8	339.3	350.8	362.3	373.9	385.5	397.1	408.8	420.5	432.3	444.1	455.9	467.8	479.7	491.7
10300	332.6	344.0	355.5	366.9	378.4	390.0	401.6	413.2	424.9	436.6	448.4	460.2	472.0	483.9	495.8
10400	337.5	348.8	360.2	371.6	383.1	394.6	406.1	417.7	429.3	441.0	452.7	464.4	476.2	488.1	500.0
10500	342.3	353.6	364.9	376.3	387.7	399.2	410.6	422.2	433.8	445.4	457.1	468.8	480.5	492.4	504.2
10600	347.2	358.4	369.7	381.0	392.4	403.8	415.2	426.7	438.3	449.9	461.5	473.2	484.9	496.7	508.6
10700	352.1	363.3	374.5	385.8	397.1	408.5	419.9	431.4	442.9	454.4	466.0	477.7	489.4	501.2	513.0
10800	357.1	368.2	379.4	390.6	401.9	413.2	424.6	436.0	447.5	459.0	470.6	482.2	493.9	505.7	517.5
10900	362.1	373.2	384.3	395.5	406.7	418.0	429.4	440.8	452.2	463.7	475.3	486.9	498.5	510.3	522.1
11000	367.1	378.1	389.2	400.4	411.6	422.9	434.2	445.5	457.0	468.4	480.0	491.6	503.2	515.0	526.8
11100	372.2	383.2	394.2	405.4	416.5	427.8	439.0	450.4	461.8	473.3	484.8	496.4	508.0	519.8	531.6
11200	377.3	388.2	399.3	410.4	421.5	432.7	444.0	455.3	466.7	478.1	489.7	501.2	512.9	524.6	536.5

Quadrant elevation table.

A previous way of doing firing tables was to use a differential analyzer, which was a predecessor to the ENIAC. On the differential analyzer, you didn't have a terribly bad job of generating trajectories. As I pointed out, the accuracy is not all that critical, when you come right down to it.

But where the problem came in was, if you wanted to calculate trajectories under standard conditions and under some nonstandard conditions, you didn't have enough accuracy in the computer so that you could take two sets of results and difference them and get a meaningful result. So to accommodate that, people went to things like adjoining systems of equations. So they first would solve the basic equation, and then they would mathematically have worked out the adjoining systems, feed the trajectories back, calculate the joint equations, and from those, be able to get the differential effects—very elegant way of doing business. If you like to read about it, Bliss's book on exterior ballistics,[30] I believe, probably goes into great discussion about the joint system and such.

[30] Bliss *(*1944).

A lot of these "pretty things" got lost. The ENIAC, again, did not accommodate that kind of calculation. It was much easier to just calculate 10,000 trajectories for all the conditions you wanted. So a lot of that kind of—what I might call "quality stuff"—got lost, at least for a while. I think it took a fair period of time before people got back out of the mentality that says the easiest thing to do is just put the simplest version of the problem on the computer and calculate the *hell* out of it.

I think that probably covers just about what I wanted to say. The whole idea of computing with the ENIAC was sort of a *hair-shirt* kind of thing. Programming for the computer, whatever it was supposed to be, was a redemptive experience—one was supposed to *suffer* to do it. And it wasn't until the 1970s that we finally were able to convince people that they were not going to have programmers continually writing little programs for them. I actually had to take my Division and sit everybody down who hadn't taken a course in FORTRAN, because, by God, they were going to write their own programs now. We weren't going to get computer specialists to write simple little programs that they should have been writing. Programming, indeed, had become a simple process, and I think to some extent, some of the earlier experience on the ENIAC convinced people that you should suffer to use a computer, whereas it had become something that was easy. [applause]

Paul Deitz:

Thank you very much, Harry. Bert Herzog has shown up, and I did thank ACM for this opportunity. Thank you very much. Frank Friedman was around the back, too, and may have slipped out. Again, our thanks to ACM. We will have these talks up on the Net as soon as possible for those who would like to do surfing. Bill Moye has copies of a wonderful brochure that he put together for those of you who want to read a little bit more about 50 years of computing in the U.S. Army.