Stories about the B5000

and people who were there

Richard Waychoff

September 27, 1979

Introduction[1]

There was a period during the early 1960s when remarkable technological strides were made at Burroughs. The following quote from Bill Lonergan's letter to the editor of *Computer* magazine expresses the feeling better than I have seen it done anywhere else. “Those were heady days in the computer field. It’s doubtful we will see their like again. I wonder if the top management of any computer company today, including Burroughs, could be persuaded to proceed with a system which included as many radical departures from current design philosophy. The reward for Burroughs gamble was a system, in the form of the B5000/6000/7000, which stayed in manufacturing for 10 years (probably the longest of any computer in the history of the field[2]) and gave Burroughs a unique architectural-based position in the industry”.

I was there during those “heady days”. This document is an effort to capture some of the stories behind the more significant developments.

Virtual Memory

My definition of virtual memory is the movement of code or data segments from a slower (secondary) memory to a faster (primary) memory so that they can be accessed more rapidly. This movement must take place without the programmer having to write any kind of instructions in his program to direct the flow of the transfer. If this definition is accepted, and I think it should be, then contrary to popular belief, the B5000 was not the first virtual memory machine. However, the concept of virtual memory and the concept of the B5000 both came from the same man and at about the same time. The two concepts became inseparable. The man is Bob Barton.

In January, 1960, I was working as a Sales Technical Representative in the Dallas District Office of the ElectroData Division of the Burroughs Corporation. John Hale was the Dallas Technical Representative Supervisor. Phillips Petroleum had been a Burroughs 205 user, but had replaced it with a much more powerful IBM 704 at the home office in Bartlesville, Oklahoma. The various Phillips refineries that were scattered over Texas and Oklahoma had IBM 650 computers. Programs were developed at Bartlesville and written in a subset of Fortran that could be compiled with IBM 650 Fortransit and executed locally.

Many of the Phillips programmers still had a soft spot in their hearts for the Burroughs 205. So when it came time for them to buy another machine they said that they would buy a Burroughs 205 computer if the following conditions were met:

A. It had to have a compiler that would compile and execute existing IBM 650 Fortransit programs with no modifications whatsoever.

B. The compiler had to compile faster than IBM’s Fortransit.

C. The time for loading the compiler and object programs had to be faster than IBM’s.

D. The object programs had to run faster.

A call was placed to Bob Barton, who was the Section Manager in charge of Automatic Programming at the Pasadena Plant (which was then considered to be the home office). Bob said that he could not spend any more effort on the 205. All of his budget was allocated for 220 projects. However, if John Hale would designate three people for the project, he would fly to Dallas for one day and teach them how to write a compiler. John selected Lloyd Turner as the project leader, and Jack Meyers and myself as the other two members on the team.

When I heard that someone was flying in from Pasadena to show us how to write a compiler, I was very skeptical. I had seen many other so-called experts from Pasadena and I was invariably disappointed.

Bob arrived with an Automatic Programmer named Toni Schumann. He spent one day with us and then went on to Washington D. C. Toni stayed in Dallas for a week and rejoined Bob for the return trip to Pasadena. She remained our contact in Pasadena for the duration of the project. She did such things as arranging for computer time and getting our programs keypunched prior to our trips to Pasadena.

The day that Bob spent in Dallas was one of the most amazing days of my life. I am sure that I never learned so much in an eight hour period. We stopped briefly for coffee in the morning and went out for a quick lunch. We did not take a break in the afternoon. The day was intense to say the least. I took Bob to the airport to catch his plane. He talked all the way to the airport and was still shouting information to me as he walked up the steps to the plane and disappeared into it. He said that IBM had spent 25 man-years on Fortransit, but that the three of us could do the job in six months.

Early in the project Jack Meyers and I started having violent arguments. Neither of us had much respect for the other. He left Burroughs for more comfortable surroundings. It took Lloyd and I nine months to complete the compiler without him. All of the goals were met or exceeded.

Bob assured us that the compatibility goal was no problem. The compile speed goal would be achieved by making the compiler a one-pass compiler. Most people believed that one-pass compilers were impossible in those days[3]. The loading speed goal sounded within reach, but he did not have much to offer as to how we would do it.

The goal of executing object programs faster than the IBM 650 sounded like real trouble to Bob. Both systems had a drum memory. The drum on the 650 rotated at 12,500 rpm compared to 3,570 rpm on the 205. However, the 205 drum was more sophisticated. It was divided into two areas. The primary storage was 80 words of 0.85 millisecond average access memory. The secondary storage was 4,000 words of 8.5 millisecond average access memory. The fast memory was accomplished by placing the read head one tenth of the way around the drum from the write head. Thus anything that was written was available one tenth of a notation later.

The average access time for the entire memory of the 650 was 2.5 milliseconds. Bob said that it seemed to him that our only chance of meeting the object speed goal was to figure out an “automatic blocking algorithm”. I did not hear the term “virtual memory” until several years later. By an automatic blocking algorithm, he meant moving segments of code from the secondary storage into the primary storage and executing it from there. Since the first goal was to compile existing programs without modification, I would have to do it without the programmer adding any flow directing statements to the programs.

Bob said that a lot of people in Pasadena (including Joel Erdwin), had tried without success to implement automatic blocking, but I should not let that stop me from trying. I would be the first person to try it with a high-level language. The success of the project seemed to hinge on that algorithm.

During the course of the next two months I did discover the algorithm. The next time that I saw Bob was in the Pasadena Plant in April, 1960. He was in the process of cleaning out his desk. He was leaving Burroughs as an employee and going to become a consultant to Burroughs, getting the same amount of pay, but only spending half of his time on Burroughs business. I described the algorithm to him and he became tremendously enthused. Frankly, I had not grasped the importance of the accomplishment.

Bob pretty well had the B5000 concept in mind by this time and had been describing it to Bill Lonergan's Product Planners. He completely dominated the design and the group of people.

Secondary storage on the B5000 was either one or two 32K drums and they were optional. Virtual storage was going to be available an optional basis, but it was not an essential part of the B5000.

By the time I described the algorithm to Bob it was around 5:30 or 6:00 pm. The sun was low in the sky and beaming into his office. The plant was mostly deserted. Bob said "Come with me". We marched straight into Bill's office even though Bill had someone in his office with him. Bob pounded his fist in his hand and said "Bill, the first drum on the B5000 cannot be optional. It just must be there". Bill looked startled at this verbal onslaught and said "Ok". That is all there was to it. There were no arguments and no questions asked. Just a quick "Ok". From that minute on, virtual memory was an inseparable part of the B5000.

So even though I was the first person to implement virtual memory, the idea clearly came from Bob Barton and it was Bob that extended it to the B5000.

Appendix A contains a Letter to the Editor of *Computer* magazine from Bill concerning stack machines and the B5000. I agree with practically all of what he says. However, on the subject of virtual memory, he gives Paul King far more credit than he deserves. Specifically, he refers to an inspirational meeting at UCLA that Paul attended in May, 1960. He says that shortly after Paul returned from that meeting that virtual memory was defined into the B5000. I can readily believe that Paul began to understand the concept at that meeting in May, but I know that virtual memory was clearly in Bob's mind on that April, 1960 late afternoon. The exact day could be determined by finding out what day Bob left Burroughs.

The Summer Of 1960 (Time Spent With don knuth)

The 205 Fortran project started near the end of January, 1960 and was completed in early October, 1960. That is pretty fast by any standards. However, there was another remarkable effort going on at the same time.

We spent the first two months designing the compiler. Then we coded for a month. As we completed the code, we mailed it to Toni Schumann to have it keypunched in Pasadena. In April, we were ready to fly to Pasadena and debug the code. (This was not exactly interactive programming.) Jack Meyers was on the first trip, but I think he had quit by the second trip. Lloyd Turner and I spent four and a half months in Pasadena between April and October.

In June we were introduced to this tall college kid that always signed his name with lowercase letters. He was don knuth. He had contracted with Brad MacKenzie to write a 205 Algol 58 compiler that would be a subset of the 220 Algol 58 compiler that was being written by Jack Merner and Joel Erdwin.

don claimed that he could write the compiler and a language manual all by himself during his three and a half month summer vacation. He said that he would do it for $5,000. Our Fortran compiler required a card reader, card punch, line printer and automatic floating point. Don said that he would not need the card reader or card punch, but he wanted a magnetic tape unit and paper tape. I asked Gerard Guyod how Brad could have been suckered into paying this college kid $5,000 to write something that had to be a piece of junk if he was only going to spend three and a half months on it. Gerard whispered his response to me. He said “We think that he already has it written. He probably did it in his spare time while working in the computer center at Case Institute.” I still wasn’t entirely satisfied with that answer because I was a college graduate whose first job was for $325 per month and I had just changed jobs and was making $525 per month. Besides that it was taking mortal human beings 25 man-years to write compilers – not three and a half man-months. I thought that Brad had taker leave of his senses.

There was only one 205 at the Pasadena Plant. It was primarily used to run the payroll. Lloyd and I were given top priority on the machine since real money was going to be given to Burroughs as soon as we successfully finished our compiler. Payroll had second priority and don was third. The payroll was usually run at 8:00 pm on Wednesday’s, but if we wanted the machine then, we could literally bump the payroll. On several occasions, we delayed the payroll until 1:00 or 2:00 am on Thursday. After a few weeks of bumping payroll, they complained and were given top priority with our project second. That was a big relief to me because we always worked at least 100 hours a week while we were in Pasadena, and Wednesday evenings provided some time to relax. On many occasions, we would go to work at midnight on Wednesday only to find the payroll half done and the operators out for a beer. So Lloyd and I would finish running the payroll to get our machine back. It was not exactly a totally secure system.

It seemed that don was always there, patiently waiting his turn. He knew when we went out to dinner and when we slept. Our compilers were both punched or cards and were the same size. We had written ours in STAR 0, the only assembler that Burroughs supported on the 205. It had been Dick Berman's first programming project. Our compiler took one hour and 45 minutes to assemble. The first week of don's project he spent in writing his own assembler. He could assemble his compiler in 45 minutes. We were green with envy. I am sure that don used only half the computer time that Lloyd and I used.

As the summer wore on don seemed to be losing the total confidence that he had at the beginning of the summer. The pressure was beginning to show. Then something happened that nearly destroyed him. Even though his compiler was going to be distributed on paper tape, he had been working all summer from cards. The time finally came for him to dump the compiler to tape. But it wouldn't fit. I thought that don was going to have a heart attack. It was a classic case of an irresistible force (don's intellect) meeting an immovable object (the size of a reel of paper tape). As I recall the problem was resolved by Brad allowing him to use two reels of tape.

Finally the last day of summer arrived and don had his compiler but not the manual. We were all physically and mentally exhausted. It was late in the evening when I saw don sitting down at Betty Potter's typewriter to start his manual. I had really grown to like and respect don, but I felt very sorry for him having to get over this last hurdle. So I sat down with him and proofread the pages as they came out of the typewriter. It seemed that he was composing and typing as fast as I could read. By morning the manual was done. Years later when don's first book came but, he told me that if I would proofread it for him that he would pay me a dollar for every error that I found, including typographical errors. I took the offer as a great personal compliment at the time, but I think that he probably made that a standing offer to anyone.

There were naturally comparisons made between the two compilers. don's Algol compiled 45 cards per minute. Our Fortran compiled 40 cards per minute. (The IBM Fortransit that we had to beat ran 10 cards per minute.) The Fortran object code ran somewhat faster than the Algol object code. There were some bugs in don's compiler that he fixed at the Christmas break. The most remarkable thing about our compiler is that I think that there was only one bug in it when it was released. I spent some time at the C. F. Braun Co. showing them how to use it and they never found any bugs. Lloyd spent some time in Texas showing people how to use it. He may have found some bugs, but I am only aware of one. It was in my code and I flew to Texas and fixed it. The patch used the last available word of memory. The machine had 4080 words of memory, and we used every one of them. I could never tell which compiler was the best, but don's compiler was certainly no piece of junk. I suspect that he had it well thought out by the beginning of the summer, but I am sure that it was not already written when he get there. Score one for Brad.

don returned to Case Institute for his Senior Year. At the graduation ceremonies, they were handing out the diplomas in alphabetical order. But they passed over don when they got to the k's. (Maybe lowercase k comes after Z). After all of the diplomas were handed out, they asked don to step up on the platform. They said for the first time in the history of Case Institute, they were conferring a Masters Degree on a student that had been pursuing a Bachelors Degree. don had taken nothing but graduate level math courses for the past two years and he was always the best student in the class. He got his Ph.D. from Cal Tech in a remarkably short time while working up to forty hours a week at Burroughs (in violation of

Cal Tech's policy that limits the number of hours that a Ph.D. candidate can work). It was apparently so easy for him that he hardly seemed aware that he was in college. He was always well rested and never rushed for anything.

In 1961, the National ACM meeting was held in Los Angeles. The keynote speaker was Tom Watson, the Chairman of the Board of IBM. Bob Barton was the second or third speaker after Watson. don, Lloyd, and I were in the audience of approximately 1,200 people. Barton was his usually charming self. He said "You people have come here to learn about programming, but I came here to tell you that I am going to put you out of business with the introduction of superior languages and superior compilers. There are only three people in this room that really know how to write a compiler and I would like for them to stand up now. They are don knuth, Lloyd Turner and Richard Waychoff."

So teacher, that is how we spent the summer of 1960.

The Algol Syntax Chart

At the beginning of the B5000 Algol Project, Lloyd and I did not know where to begin. We just sat around staring at each other for about a week. Our Manager, Brad MacKenzie sensed that something was wrong, so he brought it Barton to work with us for three weeks. Bob joined our discussions, but he could not give us a direction either. So the three of sat around staring at each other for another week.

The main thing that we were doing was wearing out our copies of the *Report on the Algorithmic Language ALGOL 60*. We were thumbing back and forth and getting nowhere. Finally, I said "My problem is that I need to see this language laid out before me, all in one place. Let's draw a flowchart of the language." Bob said "No it will not work. This is a recursive language and all you would have is a few boxes with lines leading back into themselves." He even drew a single box on the blackboard with a lot of lines running out of it and back it again.

Due to my great respect for Bob, I would never question anything that he said. So I just let the subject drop. So we thumbed through our reports and stared at each other for two more days. Then I brought up the same subject again. But this time I got up to the blackboard and tried to express myself with chalk. All of a sudden Lloyd said "I see what you are talking about". He was as excited as I was, but he had to run home for lunch. When he came back he was really excited. He said that the only thing that I had been missing was that we needed two shapes of boxes. One meaning that the metalinguistic variable was defined here, and another meaning that it was used here. Lloyd used a rectangle and a diamond for the two shapes.

At that point Barton agreed that we really had something and he went away. Lloyd and I diagrammed the entire language on his blackboard that afternoon. Then we called in our newly assigned documenter, who was Warren Taylor, and showed him the blackboard. Warren immediately liked the idea and got a draftsman assigned to work with him. They slaved for days trying to achieve the best possible arrangement of the *boxes and pickles* as Warren called them. It was Warren's idea to add the circles to indicate the basic symbols in the language. I do not know for sure, but I think that Warren decided to use the coordinate system.

Finally, the chart was complete. I have no idea who the draftsman was, but Warren would probably remember.

Automatic Programming was still in Marketing at that time, so it did not take long for our sales force (Ralph Pearson by name) to get wind of the fact that we had finally done something to make Algol understandable to the masses. Some kind of chemistry took place that I did not understand, and all of a sudden, we were flooded with syntax charts of every size, shape and composition. We had three foot long wall charts, plastic syntax charts that would fold up and fit in our pockets, and eight and a half by 11 inch charts that we could lay on cur desks. Every programmer in the section (which had grown to 15 people by then) received all forms of the chart.

Nearly everyone received the charts enthusiastically. I suspect that even the Burroughs Automatic Programmers had been as bewildered by Algol as the rest of the computing community. Ken Meyers, whose office was directly across the hall from me, showed his true technical acumen at that early age by taking Scotch tape and carefully constructing a wastebasket liner out of his wall chart. He sat it in the middle of the table in his office about 10 feet from my door for all to see.

With the advent of the syntax chart, the ice was broken and the B5000 was moving again. Recursive Descent was a small step after a few hours of gazing at the chart, and our sales force (Ralph) felt that he had a tool that would really sell the concept.

That summer (1961), Bob and I went to the University of Michigan Engineering Summer Conferences. All of the greats of the computer world were there. To list a few, there was Newell (of Newell, Simon and Shaw); Galler, Graham, and Arden; Alan Perlis; Phillipe Dreyfuss (from Bull in France); Bob Bemer (from IBM); Bob Rosen; and Alston Householder. Alston Householder was such a big name in numerical analysis that I had him ranked in my mind with Aristotle. I literally thought that Alston Householder had died somewhere around the sixteenth century.

It was somehow decided that all of these great minds should get together for a discussion one evening after the conclusion of the day's classes. Barton was obviously invited. Somehow I had felt all along that he was kind of unimpressed by the syntax chart. Probably because he had told me that it was impossible. In any case, he asked me to come to this meeting with him. Even though my ego was pretty high, I was frankly scared to go into the room with those people. I think I practically held Bob's hand as we walked in.

Almost as if it were choreographed, each person along the table made some kind of opening statement. When it became Barton's turn, he started talking about the syntax chart. As I recall, I was hiding under the table and peaking over the edge. Bob said that I had brought along a copy (at his direction) in case anyone was interested. That immediately broke up the meeting. Everyone was advancing on me to see the chart. I found myself shaking hands with a 400 year old man (i.e. Householder).

Alan Perlis was the editor of *The Communications Of The Association For Computing Machinery*. He was also one of the original 13 architects of Algol (which places him somewhere above the 12 Disciples Of Christ in my mind). He was more enthusiastic than anyone. Alan immediately started referring to it as *The Algol Roadmap*. He said that if I would prepare it as a paper and submit it to him that he would publish it in the *Communications Of The ACM*. Warren did all of the work. It was published in September, 1961. It was undoubtedly the first centerfold for any magazine other than “Playboy”. To my knowledge, they have not published anything in foldout form since then. Appendix B is a reprint of the paper as published in the *Communications*.

Two days later, Bernie Galler invited me to be one of three people in a panel discussion entitled *Whither Algol*. The other two people were Bob Graham and Alan Perlis. There were 200 people in the audience, including all of the great people mentioned before. Galler, Graham and Arden where nearly through with the implementation of the Michigan Algorithric Decoder (MAD). Graham was convinced that it must be a better language than Algol, though much to my surprise, he did not know Algol very well. Perlis made a statement that shocked me. He thought that Algol was a fine language for publishing algorithms, but not of much value as a programming language. I forgot my station in life for a few minutes and told him that it was a great programming language. He responded by saying “That’s fine for you to say, because you have the B5000 and the Algol Roadmap, but what about the real world?”.

For several weeks after the University of Michigan trip, I received requests from all over the United States for copies of the Syntax Chart. Galler, Householder and Bemer were among those asking for large quantities. The distribution of the *Communications Of the ACM* amounted to about 7,000 copies. I found that Burroughs marketing division (Ralph) had printed around 11,000 copies in various forms. At that time, there were about 14,000 programmers in the world. So there was more than one copy for everyone.

A few years later, I met Lon Grace at a PL/I meeting. He was in charge of programming languages for RCA. He told me that he had never been able to understand Algol until he saw my chart. He then offered me a job with a 40 percent salary increase. He had practically no knowledge of my background except that I had done the syntax chart.

Niklaus Wirth was at Stanford for two years during this period. Stanford was my favorite B5000 customer. They used the syntax chart in their computer science classes. I was gratified to see that Wirth used the notation in his *Pascal User Manual And Report*.

Recursive Descent

Recursive Descent is a parsing technique that employs a procedure in the compiler for each metalinguistic variable in the language. In Algol for example, there is a metalinquistic variable called a program. There are others called blocks, statements, expressions, and primaries. One form of a statement is a block. A block is the word **begin** followed by some declarations which are followed by some statements and it is finally terminated by the word **end**. A Recursive Descent compiler will have a statement procedure and a block procedure. When a statement is seen, the statement procedure is called. It may determine that the kind of statement that it is being asked to process is a block; in which case it will call the block procedure. When the block procedure finishes processing the declarations it will call the compound tail procedure that iterates through recursive calls on the statement routine. That is the recursive part of Recursive Descent. The Descent part comes from the initial assumption by the compiler that it is going to compile a program. So the program procedure is called. The program procedure determines whether it has a block or compound statement and calls the appropriate procedure. Eventually we work our way down to the primary routine.

Ned Irons preceded our invention with a Recursive Ascent technique that starts off by calling the primary routine, which makes the assumption that it should call the expression routine, which makes the assumption that it must be in an assignment statement and calls that routine. Obviously, expressions appear in places other than assignment statements. So his technique makes mistakes. It recovers from the mistakes by leaving tracks that allow it to find its was back to where the erroneous assumption was made. It then makes another guess and starts working its way up the syntax chart again. Ned's technique is obviously slower, does not exercise as tight control, and gives pretty poor error messages. He implemented his parser on a CDC 1604 at he University Of Princeton in 1960. He was working with a group from the University Of Pennsylvania.

By March, 1961 Lloyd and I had our Algol Syntax chart and a pretty good understanding of the B5000 concept. But we were still thinking in terms of a conventional compiler. That meant that something identified what kind of statement it was seeing and gave complete control to the routine for that kind of statement. That routine did its own scanning, read it own cards, printed what it felt like printing, and wrote directly into the code file whatever it felt like writing. All statement routines were autonomous.

The B5000 had a feature called Character Mode that Barton likened to the 205 and 220 Cardatron editing bands. He felt that having them built into the central processor was an advantage over having them separate from the processor. Most people were not interested in Character Mode, but I was. I am sure that I knew the instruction set in Character Mode better than anyone else.

One day I was trying to figure out what kind of role that Character Mode could play in our compiler when I stumbled on the idea of the now familiar organization which can be depicted by the three following boxes.

```
    *************      *************      *************
    *           *      *           *      *           *
    *   SCAN    *------*   PARSE   *------*    EMIT   *
    *           *      *           *      *           *
    *************      *************      *************
```

Lloyd did not especially like the idea, so I had to figure out some way of selling it to him. We spent several hours a day poring over the syntax chart. The separation of the three functions allowed me to think more clearly in terms of each function without the others clouding the issue. As the message from the chart began to sink into us, I realized that I had my sales pitch. So I laid it on him again and he bought it very quickly. There is no way to say that Recursive Descent is my invention or Lloyd's invention. It is simply our invention. It came to us at the same moment while we were working together in our secret room that we used to hide from the hoards of people that were always after us to explain the B5000 to them. All I am saying by the previous discussion is that this is the line of thinking that led me to discover Recursive Descent. I am confident that Lloyd got to the same point at the same moment by a different path.

There is another sidelight to this story that has bothered me for the last 15 years. The only hope that I have of resolving this problem is hypnosis, and I doubt that it would help either. But to be complete this part of the story has to be included.

In November, 1960 I was in California waiting for Lloyd to play his game that led to him getting his $500 per month salary. (This full story is told in the chapter called *The B5000 Algol Project*.) There was a meeting in Washington D.C. called *The National Compiler Symposium*. I was clearly the leading compiler expert in Automatic Programming because I was the only person in the group who had ever written one. So Brad MacKenzie sent me. Algol was still considered by 99% of the computing community to be impossible to implement. I knew that it was possible to implement because Bob Barton told me so. He only forgot to tell me how.

The meeting lasted two days. The focus was on how to solve the various individual problems that were posed by Algol. Most of the papers were presented by an obnoxious group of students from the University of Pennsylvania. When anyone else was presenting a paper they would laugh out loud, which clearly upset everyone else in the room. The most obnoxious of all was Peter Ingerman. The only civil member of the group was Ned Irons, and in fact he had the most interesting paper that was presented for the entire two days. It was his Recursive Assent parsing algorithm. (Of course no one called the algorithm by that name. If they had, Recursive Descent would have been obvious to the most casual observer). It was this group of students that coined the phrases that included the word *Thunk*.

I felt like I already knew almost everything that was said except for Ned's paper and one other. That was a paper by Dr. A. A. Grau from Oak Ridge, Tennessee. Frankly I did not understand one word that he said. Many years later, after Lloyd and I had been telling the entire world that we had invented Recursive Descent, Bill Price came to me and asked when we had invented it. I told him that it was about March, 1961. He said "Well I just ran across a paper by a guy named A. A. Grau that was presented at the National Compiler Symposium in November, 1960 and he describes Recursive Descent in great detail". So I apparently heard Recursive Descent described about five months before I "invented" it. Even though I did not understand one word that Dr. Grau said, I will always wonder what role it played in my subconscious.

Burroughs used Recursive Descent to write all of its compilers until 1969. Then a consultant named Bill McKeeman talked to us about a compiler generation technique. All that was needed was a compiler generator and a very tight syntactical description of the language and this thing would pop out a compiler for us. He had written two PL/I-like compilers called SPL and XPL using the technique at U. C. Santa Cruz. They ran on an early version of the IBM 360. He claimed a compilation speed of 40,000 lines per minute.

We could not ignore a breakthrough in technology like that. So Joel Donnelly was assigned the task of writing B6500 Fortran using McKeeman's technique. I was philosophical about it. As proud as I had been of Recursive Descent I felt that it was probably time for something better to come along.

A couple of strange things happened. First, Joel started saying that he was not comfortable with McKeeman's technique. Finally he flatly said that it would he far simpler and better to write a Recursive Decent compiler. Ben Dent felt that Joel was simply having trouble learning McKeeman's technique, so he told Joel to stick to the plan. The other thing was that I had a fellow on my project named Bob Novak. Bob was the only person in my group that knew anything about the IBM 360. I felt that my people should have the benefit of the knowledge of the new technique, so I got the listing and was describing it to them. Novak looked at the compile time and said "Wait a minute. He is not compiling 40,000 lines per minute. HP is compiling 4,000 lines per minute." Bob was absolutely correct. We had estimated compilation speeds of 10,000 lines per minute on the B6500 using Recursive Descent. The 360 that McKeeman was using was a faster machine. So we had launched an entire compiler project it the wrong direction because we had believed in what McKeeman told us without properly questioning him. So Ben told Joel that he could go back to Recursive Descent if he wished and everybody, including Recursive Descent, lived happily ever after.

One other thing that I should clear up. I have probably left the impression that I thought that Peter Ingerman was obnoxious. Several years later, I spent a considerable amount of time with Peter working on a PL/I standardization effort. I was representing Burroughs and he was representing that colossus of the computer industry, Westinghouse. Whereas I had thought that he was obnoxious as a college student, his increasing maturity clearly demonstrated to me that first impressions are usually pretty accurate.

The Define Declaration

The Algol project was going well by June, 1961. don returned from Case Institute. He never mentioned the fact that he had received a Masters Degree instead of a Bachelors Degree. I learned that story from a fellow student of don's named Joe Speroni, who was at the ceremony. Joe was convinced that anything that did not come from Case could not be of much value. He was a strong supporter of don, but somehow gave the credit for his intellect to Case. I suppose that it made him feel that people would view him in the same light as don. He forever endeared himself in Lloyd's heart by slipping peanuts in Lloyd's Jack Daniels and Coke at a party one night.

I was happy to see don again and launched into a description of the Algol Syntax Chart, Recursive Descent and the separation of the three functions; scanning, parsing, and emitting. don was favorably impressed. I was especially proud of my knowledge of the B5000 Character Mode. It had led me to an organization of the symbol table that seemed much better than the other organizations with which I was familiar. So I described that with more enthusiasm than the other subjects. don put the index finger of his right hand to his lips, closed his eyes, and went into hyperspace for about 30 seconds. When he came back, he said, "With that organization of a symbol table, you can allow one symbol to stand for a string of symbols". I thought that that was the best idea that I had heard in a long time. I carefully put together my arguments and all of the necessary details and presented it to Lloyd. He thought that it was a good idea. So it was immediately a part of B5000 Algol.

About a year and a half later, after the Algol group had grown to its full size, came the most difficult five days of my life. I think of it as the great debate. Someone (Dave Dahm[4] as I recall) thought that the *DEFINE* should be deleted from the language. Dave and I seemed to be on the opposite side of every argument. Fred Gerbstadt always supported Dave. Bobby Creech would be on whichever side that he thought was right.

We were asked by Lloyd to get together in a room in the basement of the Pasadena Plant to discuss the *DEFINE*. Lloyd was there less than 50% of the time. It was always a two on one situation, with Dave and Fred on one side and me on the other. Bobby was there acting as a buffer, but I cannot remember him contributing very much to the discussion.

I was trying to preserve the *DEFINE* and Dave was doing his best to get rid of it. He found some cases where the *DEFINE* became nonsense. So we invented the *STOPDEFINE* Boolean.

Dave was still not satisfied. The discussion became a fight. After four days of fighting Dave and Fred, I was exhausted. Lloyd usually got to work at about 8:15 am. I was sitting at his desk, when he arrived. I said "Lloyd, I am exhausted. This argument over the *DEFINE* is destroying the project. The *DEFINE* is a good concept. I am appealing to you as a friend to call these wolves off my back and leave the *DEFINE* alone".

Lloyd said "That is good enough for me". We walked together to that room in the basement that had become a torture chamber for me, and Lloyd said "The meeting is over. The *DEFINE* stays in the language as it is."

There are clearly problems with the *DEFINE*. It was intended to make programs more readable. But I have seen cases in the SDL[5] compiler where its use has made the code almost unreadable. One might think that if Lloyd had allowed the argument to continue, that a better *DEFINE* would have resulted. But I do not think that would have been the case. Our arguments were becoming less and less objective. The result would have been that we would have lost our ability to work together.

A year or so later, Dr. Edsger W. Dijkstra came to Pasadena to see our B5000 Algol. I decided to describe the *DEFINE* to him. Whereas I had withstood the slings and arrows sent my way by Dave and Fred for four days, Dijkstra tore me apart in about four minutes. His objection to the *DEFINE* was its scope, which begins where the declaration appears and ends at the end of the corresponding block. He did not make any attempt to tell how to make a better *DEFINE*. He simply thought it was a terrible idea.

Several years later Dave thought of the parameterized *DEFINE* and implemented it in a compiler called XALGOL, which was a cross between B5000 Algol and B6500 Algol.

B5000 Algol Stream Procedures

Stream Procedures are a language feature that was developed to allow B5000 Algol to generate code that was executed in the Character Mode in the B5000. I was introduced to the Character Mode by Gerard Cuyod who described it by saying that it was just like 205 and 220 Cardatron.

Cardatron was the interface between Burroughs 205 and 220 computers and IBM tab equipment. We used an IBM 407 for our line printer, an IBM collater for our card reader, and an IBM reproducer for our card punch. Cardatron had a small drum memory that rotated at 22,500 rpm. The drum contained space for data and editing information. The editing information was encoded in strings of decimal digits. These strings were contained on format bands on the drum. A digit punched in some particular column of the card, or in some location of the output buffer indicated which formatting band should be applied to this string of data. Dick Berman wrote a format band generator that eased the pain of writing these things. To code the format bands, one had to think of a line as being a series of ten digit fields and then code each field from right to left. Compared to Cardatron, the B5000 Character Mode looked pretty good.

There was this fellow named Bill Logan who had been an Automatic Programmer on the 220. When the B5000 project began, some of the Automatic Programmers that showed an aptitude for architectural design were transferred to Bill Lonergan's Product Planning group. Logan wanted to be transferred, but he wasn't invited. So he came in one weekend, dragged his desk about 100 yards from Automatic Programming to Product Planning and there he sat. I guess that neither Brad MacKenzie nor Bill Lonergan felt strongly enough about the subject to tell him to move his desk back, so they eventually transferred him on paper. I remember that Bill was married to a lady whose parents were Japanese, but her only language was English. Her father came from Japan to visit her during this time. Unfortunately, his only language was Japanese. Bill said that his wife and father-in-law just sat around and looked at each other for about two weeks, and then he went home.

The reason that I mention Logan is that he is the guy that designed the B5000 Character Mode. It was considered to be the least interesting part of the B5000 and I guess that they couldn't figure out what else to do with him, so he got the job. (Logan made another contribution to Burroughs by introducing Bob Barton to Ken Meyers. Bill and Ken had worked together at a Chicago based insurance company).

Now if one considers the origin of the notion of the Character Mode (i.e. Cardatron), and the nature of the person that designed it (Logan), and then considers the origin of Algol 60 and the nature of its designers, then a better appreciation of the task of marrying the two might be gained. Lloyd and I were dumbfounded. Complicating this task, was that nothing was known to anyone about character manipulation languages.

The most ambitious of all of the 220 Automatic Programming projects was the Algol 58 compiler. I am sure that it was the best compiler in the world in 1960. It was written under Barton's direction by two outstanding software designers. They were Joel Erdwin and Jack Merner. Joel left at the conclusion of the project to assume a very important position at Computer Sciences Corporation. Jack was invited to join the Product Planning group and accepted. He made an excellent contribution to the design of the B5000 stack mechanism by figuring out how to combine three stacks into one[6] (contrary to what Lonergan's letter to the editor of *Computer* says).

Jack and I lived in Azusa which is about 25 miles from the Pasadena Plant, so we decided to alternate the driving chore. I don't remember any morning when Jack either picked me up on time or was ready when I got to his house. He decided that he could get ready faster in the mornings if he left his tie tied and hanging on the doorknob of the front door (he was always slipping it over his head as he rushed to my car), and if he stopped brushing his teeth. Neither of those measures seemed to help. When he was driving he would leave work about an hour early and go out for a few beers. Often times when he wasn't driving he would leave work early and go out for a few beers. So it was always fun trying to find Jack when it was time to go home.

Jack had worked long and hard on the 220 Algol project and had made his contribution to the B5000 architecture. Now he was at loose ends. During our trips to and from work he frequently offered to help us with any problems that we might have on the B5000 Algol project. We had only consulted with him one time, and that was with some problem that we were having with the <for statement>. He came up with the answer almost immediately. It was obvious that he was feeling bad about being left out of the action.

After Lloyd and I struggled unsuccessfully for several days trying to design a language that would marry the Character Mode to Algol 60, we thought of Jack. He seemed delighted to be a part of the effort again. We did not make any attempt to work with Jack. He just took the problem and went back to his office in Product Planning. I think that don probably worked with him to some extent on the problem. I know that they were working on their highly controversial *Algol Confidential* paper at that time.

After about two weeks, Jack came back. He was clearly less than delighted with what he had produced. In fact, he was apologetic. Lloyd and I thought that the language was awful. Jack thought that the language was awful. So we jointly tried to improve it, but we simply could not think of anything better. Finally, we just gave up and accepted it.

Not all of the stories in the big city are happy stories. I think that this is the unhappiest of them all. Stream Procedures were clearly the biggest blight on the B5000 Algol compiler, but we just did not know how to do anything better.

Dollar Sign Cards

The invention of Dollar Sign Cards came about as a result of Burroughs determination to get away from a dependence on IBM tab equipment. Burroughs magnetic tape equipment was far superior to anything that IBM had. The 220 tape system had 19 commands. For example, one could position the tape to a record that contained an arbitrary string of symbols. The positioning could be done in either direction. One of our tape devices looked like a coffin. It was called a *DATAFILE*. It had 50 lengths of tape with two eight-digit tracks of information on each tape. So logically, there were 100 tapes in this thing that was exactly the size of a coffin[7]. Every record was addressable. There were no reels in it. The tape just piled up on one side or the other of the pinch-rollers. The biggest problem with it was that if a strip of tape was left to lay there for more than a week it would stick together. Then when someone tried to read it, the whole wad of tape would come up and get all tangled up in the pinch-rollers. So any good Field Engineer would have a hip-pocket program that moved all of the tapes every day.

Our printer was a joke. It was a huge off-line thing that was bigger than a B1800 with a line printer and a couple of pack drives. It was used by writing a tape on the 220 and carrying it to the printer subsystem. It was the first transistorized thing that was ever built in the Pasadena Plant. Someone (probably Roy Guck[8]) told me that it used so little power, that when the room was really cold, that it would work with the power off. I guess I have always believed that until I startled writing this chapter this morning.

The printer was the first drum printer that I had ever seen. It would print 1,500 lines per minute alphanumeric, and 1,800 lines per minute when only numeric values were being printed. It cost $243,450. Its reliability was somewhat suspect. I walked through the room that it was kept in every day for a year, and the only time that I ever saw it working was at a trade show in Los Angeles. The main reason that I went to the show was that I heard that the printer was there and working. I suspect that the printer was a strong contributor to the demise of Toni Schuman's career with Burroughs. Doug Bolitho was giving a plant tour to a group of potential customers one day and he somehow had the printer printing something. Toni walked into the room and loudly exclaimed “My God it's working”. She left Burroughs shortly after the incident.

Our card equipment was non-existent. Some engineer got the idea that reading cards was just like reading bank checks, and Burroughs clearly had the world's best check sorter. It would read 1,580 checks per minute. It had a rotating drum with holes in it and a vacuum tube up the middle. When that check got near the vacuum drum it would suck it against it and fling it up the slot at about 400 inches per second. It was really nice. However, the engineer didn't seem to understand that programmers had the habit of punching holes in cards, and that is severely detrimental to a vacuum feed system.

I did see the card reader work on dozens of cards at a time. But as soon as it would hit one with a lot of holes in it, it was all over with. Of course Doug Bolitho could make it work. I don't know what his job was, but he gave a lot of plant tours to potential customers. He had some kind of magical quality whereby he could walk up to a machine that was covered with cobwebs and dust and turn it on and that thing would work, even if it had been broken for years. He could put Uri Geller to shame. There were times that I would be standing there cursing the card reader because it would not work when Doug would walk in with one of his plant tours. He would grab a handful of cards and twist them and bite them and step on them; then he would put them in the card reader and it wouldn't miss a beat. The entire Algol group would gather around his demonstrations to try to figure out how he did it. As I recall, he would start with a handful of cards straight out of a new box. So there were no holes in them.

The B5000 Algol was so much faster than any other compiler in existence, that it was viewed as a program loader instead of a compiler. The IBM mode of compilation was to punch their Fortran programs in cards, then go card to tape on a 1401, then mount the tape on a 7090 and compile from tape. As they compiled, the object code was punched into binary decks. Since compilation was so slow and expensive, program maintenance consisted of changing the binary decks. The source code usually got out of date pretty fast.

We were so dedicated to hiding the object code from anyone that might want to tinker with it, we were only going to allow it to exist momentarily while it was actually executing in the machine. Our view was that magnetic tapes were things that held data, not programs. We laughed at IBM for making their customers buy a 1401 just to use Fortran.

Our view started changing when we found out how unreliable the card reader was. As our compiler grew to 2,000 lines, it became nearly impossible to compile it. So we changed our philosophy. We wrote a program that put our compiler and test cases on magnetic tape and we started compiling them from there. However, any time that we wanted to change anything, we would have to modify the card decks again and go through the pain of reading the cards over and over again until the card reader finally read them all correctly.

Compounding the problem was the fact that the MCP did not know how to rewind a tape. So we could programmatically pick up the I/O descriptor and turn on the rewind bit and perform a release on it, or we could walk over to the tape unit and manually rewind it. Sometimes the tape unit would not rewind no matter what we did to it.

One day only Bobby Creech and I were at work. That was really unusual, because everyone was usually there all of the time, seven days a week, except to grab a few hours sleep. Bobby and I decided to go to the Antique Inn and drink beer. They had these huge cut glass steins of beer for 50 cents. We started discussing the rewind problem and within minutes mapped out the notion of only reading the cards that had changed and merging them right in the compiler. We even decided on the a special use of the $ to tell us if the compiler should compile from cards only (i.e. $ CARD) or compile from tape while merging patch cards (i.e. $ TAPE).

The scheme implied using part of the card for sequence numbers, which irritated some people. We decided on columns 73 to 80 for the sequence numbers. The other members of the Algol group liked the idea, so I coded the world's first *READACARD* routine. It worked like a charm and I was very proud of it. It noticeably speeded up our efforts.

Then one day Brad MacKenzie came back to the D Lab to see how we were getting along. I said great, and told him that we had solved the card reader and magnetic tape rewind problems. As I proceeded to explain the solution to him, I could see that he wasn't buying the idea. Brad was almost certainly conditioned by the criticism that our software had received because we had to use an extra card column to designate which Cardatron format band to use. Now I was proposing the use of eight extra card columns. He just would not buy it.

To that point, I had never had to convince Brad of anything. Every decision that was made had been made by Lloyd. So I didn't like having to argue with Brad, even though he was my manager. But I explained to Brad that it would significantly slow our progress if we had to junk *READACARD* and the dollar card notion before the tapes and card readers started working. So he said that we could leave it in until they were straightened out, but he didn't want any customers to have to put up with not being able to use card columns 73 through 80. Fortunately, for the dollar card notion, our tapes and card readers never did work properly and Brad never enforced his order[9].

The strangest bug that I have ever seen involved dollar cards. It is well known that an improperly written *DEFINE* can wreak havoc in a compiler. Dave Dahm got the bright idea of limiting *DEFINE* strings to 2,048 characters so that we could assume a missing # after that time and give up.

One day we were compiling the compiler and the tape was ticking along at its usual rate, with an occasional card being read. Then suddenly the tape stopped and the card reader took off and we got a whole flurry of meaningless syntax errors. It took us quite a few hours to track down the problem. Someone had left a # off of a *DEFINE*. The compiler counted up to 2048 characters and gave up. The place that it gave up was in the middle of a big comment in the *READACARD* routine where I was discussing the use of the control card "$ CARD". That $ was exactly the 2,048th character after the beginning of the *DEFINE* string. So the compiler stopped merging cards and tape and began compiling from cards only.

The B5000 Algol Project

Barton continued to be the most powerful force behind the B5000 even though he was on the outside, looking in, and only spending half of his time with Burroughs. I had spent many hours with him, essentially sitting at the feet of the Master. When he asked Lloyd and me to move to Pasadena to work on the B5000 Algol project, I jumped at the chance.

But Lloyd played it cool. He talked about everything from how much he loved Dallas to how good his wife's job was. He finally said that he would come for $900 per month. I thought that was the most obscene thing that I had ever heard. I tried to point out to him that he was blowing a great opportunity, but he would not change his mind. I am pretty sure that $900 per month was more money than Barton was making from Burroughs.

I moved to California in October, 1960 and started studying Algol and the B5000. My arrival brought the size of the Automatic Programming Section to 11 people. That seemed more than enough to do any meaningful job. Brad McKenzie was my Section Manager. Other members were Dick Berman, Ken Meyers, Molly Chalkley, and Larry Sturges.

I have loved California from my first trip until today. Bob Barton and Bill Lonergan and I played tennis at least 45 Sundays out of the first 52 that I spent in California. We never once had to cancel out because of bad weather. We always played doubles. Our fourth varied from one Sunday to the next. It was frequently Don Stevens or Bob Forrest. When Lonergan was lured away from Burroughs by Univac (they increased his salary from $20,000 per year to $40,000) Barton and I started playing singles about four or five times each week. I always felt that I was better than Bob, but he always won. I tried everything to beat him. One day I did some arithmetic and figured that Bob and I had played over 2,000 sets of tennis. I had won four. Bob was getting sadistic. He used to taunt me on the tennis court, saying things like "Come on, can't you do better than that". I would get so damned mad at him that I could have killed him. One day in particular I was really seeing red. My serve was the best part of my game. I decided that I was going to hit him right between the legs with my next serve. I hit that serve so hard that I know he never saw it. It hit its mark. He dropped like a shot. I said "Do you want to play or not". One night he called me from Indianapolis and asked me if I would play tennis with him the next day. I said that I was tired of getting beaten and that it wasn't any fun anymore. We haven't played since then.

I was convinced that I had seen the last of Lloyd. But by some set of circumstances that were incomprehensible to me, they actually offered Lloyd $900 per month to move to Pasadena and be the B5000 Algol Project leader. I remember thinking to myself, "My God. Just think of the responsibility that he is taking on by accepting that much money."

Lloyd arrived in Pasadena in January, 1961. We were still enjoying the success of the 205 Fortran project. My feeling was that if we could write a Fortran compiler in nine months that we could damn near walk on water. With the B5000 we had a beautiful machine designed expressly for a beautiful language. We even had Barton around to help out if we needed him. To put it bluntly, we were in Fat City. But a strange thing happened. We did not know where to begin. We were finally able to get the project going with the inventions of the Algol Syntax Chart and Recursive Descent. Separate chapters of this document are devoted to those two topics.

One of the most important lessons that Barton had taught me during that amazing day when he and Toni came to Dallas, was that the quality of a program comes from its design. Programming and debugging are things that are tacked onto the end of the project and should not occupy more than a small percentage of the time. We didn't have a B5000, but we had all of the blackboards, and chalk that we could use.

We discussed every design in detail and drew very complete flow charts. Once we had the flowcharts, we did something that I called *Playing Computer*. Basically we took many variations of the Algol 60 constructs and walked them through the flowcharts. I found a lot more bugs on the blackboard than on the computer. In later years, I heard Lloyd say that we had over-designed the compiler. But I do not agree with him, if he was sincere.

As summer approached, Barton started telling us about this bright fellow that had worked for him at Shell Development in Houston and also during the summer of 1959 at Burroughs in Pasadena. I felt that it would be a nice gesture on our part if we provided him with a summer job and gave him the benefit of our experience. His name was Dave Dahm. He had worked at Lockheed during the summer of 1960. Dave was at Princeton working on a Ph.D. His dissertation was in an area of topology called Knot Theory. Dave is the only person that I know that has seen Einstein in person.

When Dave showed up in Pasadena nearly everyone remembered him, even though it had been two summers since he had worked for Burroughs. They mostly remembered him as the fellow who would occasionally take a sharp pencil and reach up into the coffee machine and poke holes in the bottom of the cups. Dave always denied that he used to do that but his denials were never very convincing.

We showed Dave all of the things that we were doing, and we spent a very pleasant summer together. Dave was slow to accept the Algol Syntax Chart as a reliable tool, but he seemed to fully appreciate it, as well as all of the other interesting things that we were doing, by the end of the summer.

Then the fourth member of the group arrived. It was Fred Gerbstadt. I had a private office, which was smaller than a two person office. When Fred arrived I had to share it with him. I didn't like the loss of status, or the fact that Fred was always smoking something, or the fact that Fred always invited some people in to play chess for an hour or more at lunch time.

Fred didn't like it much when I would ask him to leave the office when I received a call from my girlfriend. But he handled that pretty well. What he couldn't handle was that I had an Accutron wrist watch that I was always bragging about. I had paid $171 for it. Fred had an $8 Timex. He used to drive me nuts by secretly setting his Timex about every two hours, and whenever we would call the time-of-day to check our watches, his would be more accurate than mine.

From the day that I graduated from high school I always had a nice watch of some kind. Shortly after I received my new watch as a high school graduation gift, the face got all fogged up. I was really concerned, so I caught a bus and went downtown to the jewelers where my parents had bought the watch. The jeweler took it apart and cleaned it up for me.

Then he said “I’ll tell you a little secret. If it ever happens again, instead of bringing it to me, just lay it on a bare light bulb for a while”.

One day Fred came to work and he said “Damn. The face of my Timex is all fogged up”. I said “Don’t worry. Just take it home tonight and lay it on a bare light bulb for awhile, and the heat will drive the moisture right out”. I noticed that Fred was pretty quiet for the next several days. In fact he wasn’t speaking to me at all. What neither of us had taken into account is that his cheap-ass Timex was mostly plastic. The face melted down all around the hands and the strap melted clear off. I guess it smelled awful. Anyway. he just knew that I had done that on purpose to get rid of his Timex that had been the chief instrument for his favorite joke.

The fifth and last member of the group was Bobby Creech. He was known as “The fellow that Lloyd Turner used to work for, but now works for Lloyd”. Bob said that Lloyd never introduced him to anyone without telling them that Bob used to be his boss at Tempco Aircraft in Dallas. Lloyd had as much trouble getting Bob to leave Dallas as we had in getting Lloyd to leave Dallas. We knew that Lloyd really wanted Bob because he always talked so highly of him. Lloyd must have done a pretty good sales job on Bob also. When we were finally introduced, Bob said to me “My gosh, after all of the stories that I’ve heard about you, I thought that you would have at least three heads”. I was obviously very complimented.

Dave returned for a formal interview before getting his Ph.D., but I felt that it was a foregone conclusion that he couldn’t resist rejoining the Algol team. Dave wouldn’t give Lloyd a firm answer during the interview. I invited Dave to spend a Saturday afternoon at my place. It was a beautiful California day, so we sat out on my patio in Sierra Madre and got drunk in the sunshine. Dave told me that he would love to work on the project, but he wanted $1,000 per month, which was $100 per month more than I was making. I was really pleased with his admission, and I told Lloyd at the first opportunity. Lloyd asked how it would make me feel to be making less than Dave. I told him that it wouldn’t bother me at all. So the rest was a formality. With Dr. David Dahm’s arrival in June, 1962, the team was assembled.

The last flowchart in my Algol Project Manual is dated 6/22/62. So that is when the design phase ended and programming began in earnest.

An unfortunate thing happened to Dave as soon as he arrived in California. Lloyd and I always tried to stay in room 101 of the Saga Motel during the four and a half months that we spent in Pasadena during the 205 Fortran project. We almost always got that room, because there was a lot of traffic noise and no one else wanted it. But we rarely ever slept anyway so it didn’t bother us. Besides, it was the best room in the motel for parties. Lloyd wanted the best for Dave while he was getting settled in Pasadena, so he got him a room at the Saga. I believe that it was Dave’s first weekend in town that he decided to go to Hollywood Park to the horse races. All of his worldly possessions were in his room. That concerned him because he did not quite trust the people at the Saga. So he moved everything into his car and went to Hollywood Park. While he was there someone broke into his car and stole all of his worldly possessions.

My plan for writing the compiler was to write a kernel compiler in Algol, then play computer with our flowcharts and generate the object code that our eventual compiler would have generated. But that rigor did not hold up very long. I stuck to it longer than anyone else, but not all that long. We were programming in a terrible language called the Operating System Implementation Language (OSIL). Dan Brannies wrote the assembler. It ran on the 220 and generated a mess of cards that then had to be carefully arranged by hand. I think that Dave and Fred were the only ones that understood the complicated process well enough to go through it. Dave wrote some programs to help him put the mess together. If everything went, perfectly (which it rarely ever did), we could assemble the compiler in nine hours.

My area of the compiler was the scanner. It was a huge Stream Procedure with 32 parameters, carefully arranged in order. They were carefully arranged because it took one microsecond to access the first parameter, two microseconds to access the second, and so on until it took 32 microseconds to access the 32nd parameter.

It was the function of the scanner to load a table called *TABLE* with strings of values representing tokens. The parsing routines would pick up the values and do their job on them. I had envisioned that the other guys would load dummy values in *TABLE* and debug their code in parallel with my debugging. But Dave insisted that they needed their input from the real scanner, so I had practically zero time to get my stuff debugged. I felt that I was bringing the entire project to a halt.

Jack Merner and John McNeley wrote a B5000 simulator that ran on the Philco 2000. We went to San Jose to the General Electric plant to use their Philco 2000. When we had to reassemble, we would drive to Stanford and use their 220. The night shift was supposed to shut down at midnight, but Lloyd could always convince them to stay until after 1:00 am. When they left we would go to Stanford and reassemble. I averaged three hours of sleep a night for weeks. It was the most grueling time of my life. There was no letup when the scanner started working. Just constant pressure from Lloyd and ourselves.

Finally we got the word that the B5000 was working. But it was working so poorly that we were better off on the simulator. Even when we switched over to the B5000, the pressure continued. Finally, one glorious day, the kernel compiler compiled itself successfully and repetitively. We were free from OSIL at last. Our turnaround time dropped from nine hours to four minutes. Our progress accelerated tremendously, but the pressure was still on.

There was no time for anything outside of work but eating and sleeping, and darned little of that. We worked over 100 hours every week. One day in July, I was wanting to order some tickets for the Riverside Grand Prix which was the last weekend in October. So I asked Lloyd, in the front of the rest of the group, if I could take off one weekend in October. They all thought it was funny as hell for me to ask for a weekend off four months in advance. It was a good thing that I asked. I really felt guilty when the weekend came and I was the only one that didn't have to work.

There was especially no time for any outside relationships. My wife and son packed their bags and went to Tulsa saying that they were not coming back until I asked them to. I never asked, so after several weeks she came back and we were divorced. Lloyd and his wife were divorced the same month. She moved back to Dallas. Bobby's wife moved back to Mineola, Texas and I assumed that that were getting a divorce, but I don't think that they ever did. Fred's wife refused to believe that he could be spending that much time at the office, so she concluded that he was seeing another woman. I remember Fred, almost pleading with Lloyd to be able to spend more time at home. Lloyd said ok, but nothing changed. I think Fred's wife left him for a short while, but I'm not sure of that. Dave didn't have any relationships outside of work at the beginning of the project, and he didn't have time to develop any during the project. By the end of the project, our families were all back together again, temporarily. Now we are all divorced except for Bobby. The reasons for the divorces were not related to the project.

About September, 1963 the compiler was completed. It was clearly the best compiler on Earth. We were compiling a very rich language and generating good code at the rate of 1,750 lines per minute. The next fastest compiler was about 400 lines per minute on a machine that was much faster than the B5000. But our compiler wasn't good enough. All of us, especially Lloyd, knew that we could do better. My pet project of writing the scanner, in a Stream Procedure had been a mistake. It caused all of our largest tables to be non-overlayable. We also knew of other improvements. So we started all over again.

At this point, the five of us were the world's best B5000 Algol programmers. Errors were something of the past. We literally rewrote every single line of that compiler and debugged it, in three weeks. It was less than 6,000 lines long. I wrote all six of the I/O statement routines and the *MONITOR* and *DUMP* declarations. Remembering my American history, I called the *MONITOR* procedure *MERRIMAC*, with the comment, "THIS TIME THE MERRIMAC WILL HANDLE THE MONITOR". I am pleased to see that the name is still used in some compilers today. The six I/O routines required 900 lines in the compiler. I believe that I had three syntax errors (keypunch errors) and one logic error. The logic error surfaced a few days later. All of my test cases ran correctly the first time. All of the other guys had similar experiences.

So we now had our "lean and clean" compiler as Lloyd called it. So we junked the world's best compiler it favor of a much better compiler. All of our large tables were overlayable. It would compile 2,000 lines per minute and only occupied 4,000 words of drum memory. The compiler could compile itself in three minutes. Stanford and other universities used to teach courses on that compiler.

The first inkling that I had that something was seriously wrong came from Bill Conlin. Bill was one of the few salesman that I had any respect for. He had made the effort and understood the B5000. Bill made same derogatory comment about the compiler. I couldn't believe my ears. So I took him to task. It turned out, that his complaint was with the MCP, and not the compiler. I had never thought that the MCP was very good, so I agreed with him. Actually, I never thought that anything was of much value that the Algol group didn't do. I discussed Bill's complaint with Lloyd, which started him thinking. Actually he had probably thought of this a long time before on his own. It became clear that the B5000 would never be viewed in the proper light if the ALGOL group didn't rewrite the MCP. Brad MacKenzie wouldn't hear of it. He was concerned for the feelings of the MCP group. Lloyd essentially went over Brad's head and offered his boss, Dean Holdiman, a deal. The five members of the Algol group would rewrite the MCP in our spare time (i.e. nights and weekends) if Burroughs would buy us five brand new Corvettes. They cost $5,000 each. There was a period of several days when I really thought that Burroughs was going to accept our offer. I thought that Dean was fighting for it and lost.

Then along came the Burroughs head-per-track disk. It gave Brad the excuse that he needed to allow Lloyd's group to write a new MCP without hurting the feelings of the MCP group. They still had their hands full with the drum MCP. Unfortunately, Dave and I had come the point that we had lost our ability to work together. We had had too many arguments to continue.

The B5000 would have clearly failed if it were not for the disk MCP. It, in combination with B5000 Algol was a thing of beauty.

I was given the job of turning the Algol compiler over to the maintenance group while the other guys were starting the disk MCP. I still have the notes for the class that I taught, along with the time spent on each subject. The class met from 3:00 to 5:00 pm daily and started on October 31, 1963. I spent 52.5 hours teaching the class. Every procedure in the compiler was covered. Our former documenter, Warren Taylor was the section manager in charge of maintenance. His project leader for Algol maintenance was John Skelton. I only recall one time that John ever came to me for help with a problem. I have never seen any statistics on the reliability of the compiler, but I have to think that it was excellent.

We were never paid one cent of overtime in any of its various forms. Brad told me that he didn't care if I didn't do anything for three months after the project was over. That is when I took up golf. So I was on the project two months before anyone else, and stayed on it six weeks after everyone else had left it.

I cannot give enough credit to Lloyd. He had a hard-headed group of angry young men to control. There was never a doubt as to who was in charge. He made every single decision himself. Lloyd left Burroughs for a year and Bobby Creech made a very touching speech at his going away party. I wish that I could exactly recreate his words, because it captured my feelings exactly. He essentially said that Lloyd may be the most demanding taskmaster in the world. He invariably gets from his employees far more than they themselves thought that they were capable of. But in every case, when a former employee looks back on his career, he feels that the years spent with Lloyd were the most productive and in some ways, the most satisfying of his life.

The Increasing Authority From Detroit

The ElectroData Division of Burroughs had a home office. It was in Pasadena. The top man in the company was Ray Bradburn. The second top man was Ed McCollister. I had some vague awareness that we had been bought by Burroughs, and their home office was in Detroit, but it did not seem to matter. Detroit was a place that was used to tease us when we called Dallas "Big D". I thought that Dallas was a great place to live, but if I ever had to transfer to the home office (i.e. Pasadena) I could probably handle it. My number one travel goal was to cross the Continental Divide some day. I had already accomplished my previous goal of being in some state other than Oklahoma. My first ride in a commercial airplane had been during my interview trip to talk with John Hale. That was the main reason that I went for the interview.

When Lloyd and I flew to Pasadena in April, 1960 to start debugging the 205 Fortran compiler, I was somewhat overwhelmed. When Bob Barton and Toni Schuman came to Dallas in January, 1960 to teach us how to write a compiler, they flew in a propeller driven airplane. By April, jet service had just been initiated between Dallas and Los Angeles, and we were flying on a jet airplane. I never dreamed that I would get to fly on a jet airplane.

I was never aware that we might have schedules, or budgets, or a staffing level. It never once entered my mind that we might be held accountable to anyone for anything. The only goal was to produce the best compiler in the world.

The B5000 was designed to be an Algol machine. The character set included as many of the Algol characters as could be fitted into the 64 character limit. That included the symbols for equivalence, implication, logical or, logical and, logical not, and multiply. My first clue that the Algol compiler was not the only consideration for the B5000 came about when we were told that those symbols would have to be replaced with #, @, &, $, * and %.

The next time that I was made aware that we were not completely in control of our destiny was because of a visit by Dr. Edsger W. Dijkstra. Dr. Dijkstra was the world's foremost computer expert in my mind, with Barton a close second. There were only two meaningful Algol compilers in the world at the time. One was our B5000 Algol and the other was Dijkstra's compiler.

Dijkstra did not have a virtual memory machine. His compiler was held on a magnetic tape. So when his operating system sensed that it needed to compile something, the compiler was read into core memory and it did its work. The object code required an interpreter which was also on tape. The execution required the interpreter and object code to be read into memory before the process could start. The B5000 Algol compiler could compile and execute the program:

```
BEGIN
END.
```

in less than three seconds. The first thing that Dijkstra wanted to see was BEGIN END. compiled and executed.

We apologized for the "." being required since that was not a part of Algol. The "." did not bother him at all. He thought it was a good idea. Dave punched the program on cards for him and I showed him how to watch the SPO (a model 33 teletype) for the BOJ and EOJ[10]. When everything was ready, Dave hit start on the card reader and B5000 went zap and it was done. Dijkstra gave a look of astonishment and said “My God. How does one get any conception of how fast it is?”. I am sure that his minimum compile and execute time was two orders of magnitude slower.

So the worlds greatest computer expert returned to Europe and told his friends about the amazing B5000 Algol system. They decided to order three B5000s. We were very happy when we heard the news. However, Ray Macdonald was Vice President of the International Group at the time, and he decided that it would cost too much to set up a maintenance organization for just three machines. So he blocked the sale. Sigh.

Few Burroughs salesman understood the B5000 until it had been on the market for several years. Our current mechanism for getting a Project Development Authorization approved includes a marketing forecast. I do not believe that anything as innovative as the B5000 could possibly survive the tests that we put our new ideas through. Even if it survived the tests and product development started, our benchmarks seem to encourage minimal thought given to the design of our software. We see benchmarks such as:

A. Successful Clear/Start[11].

B. BOJ for an object program.

C. Simple program compiled.

This kind of benchmarking almost demands that coding starts as early as possible. I would much rather see benchmarks such as:

A. Design of the language complete.

B. Design of the Scanner complete.

C. Design of the Statement Parsers complete.

D. Start of programming.

E. Project successfully completed.

Somehow I have the impression that Detroit does not feel that we have time to produce products that are superior in innovation and quality. So I can understand Lonergan’s opinion that no company, including Burroughs, is likely to duplicate the feat in today’s environment.

Reentrant Code

In June, 1961 I attended the Engineering Summer Conferences at the University Of Michigan. One of the speakers was Phillipe Dreyfus from Bull Computers in France. They had a working multiprogramming machine called the Gamma 60. Dreyfus used the terminology *Virtual Processors*. It was the first time that I ever heard the terminology "Virtual <anything>" applied to computers. I really liked it.

He devoted an entire session to the French Railroad payroll program. The French Railroad had gone on an economy program and reduced their number of employees from 540,000 to 360,000. So that was the size of the payroll that was run on the computer in Paris. That payroll must be the most difficult payroll in the world to run. The size of the payroll is only the tip of the iceberg. An engineer's (i.e. a train driver) salary is a function of the amount of power that he uses. So if he is on an uphill run, he makes more money than on a downhill run.

The employees live in government housing and their rent is deducted from their pay. People with a lot of dependents get more money than people with a few dependents. A veteran gets more money for each medal that he won in the war. The amount of money depends on the medal. A Purple Heart is worth more than a Sharpshooters medal.

They found that the most throughput could obtained when three copies of the program were running simultaneously. They had implemented Reentrant Code. I felt that this was a really exciting concept that was directly applicable to the B5000, and in fact we could do it even better on the B5000 since it did not allow code to modify itself. Each process could have its own stack.

As soon as I returned to Pasadena, I started lobbying to get our MCP group to implement Reentrant Code. Clark Oliphint was head of the MCP group. He more than had his hands full, so he was not interested in implementing any more goodies. Dave Dahm was working on the Algol project as a summer student employee. It was the summer preceding his final year in college. So I appealed to Dave and Lloyd for support. But I could not convince them that Reentrant Code was worthwhile, so I gave up.

At the conclusion of the B5000 Algol project, the other four members of the team went on to write the disk operating system. That operating system implemented reentrant code.

My Debriefing By Barton

From October 20, 1961 until January 11, 1962 I was a closet consultant. Barton had been away from Burroughs for a while, but was back again as a consultant. He hired Lloyd and I to work nights and weekends for the same hourly rate of pay that we were making at Burroughs. I worked 74 hours during that period.

The main thrust of the effort was to find out what problems were left in the B5000. Barton had been the major influence on the machine during its design phase. By this time, Lloyd and I had worked with it for a year, so now it was time to find out what kind of problems we had encountered; get them corrected; and start on a new machine. After all, everyone knew that the life of a machine was three years and it took two years to design a new machine and get it on the market.

The problems that we discussed were:

A. The B5000 implementation of Algol had a major restriction on the published version of Algol 60. Simply stated if a procedure was local to a procedure, the inner procedure could not directly address variables that were local to the outer procedure. We called this the uplevel addressing problem.

B. The Character Mode and Stream Procedures were terrible. We still did not know how to do string manipulation.

C. Iteration left something to be desired. Nearly everything in Algol 60 “said something” to me about machine architecture. The left me blank.

D. Input/Output was merely state of the art. Barton wanted an I/O system that would perform off-line processes without interrupting the control processor.

E. Barton wanted more lookahead built into the architecture.

F. He wanted to do away with queues for controlling concurrent processes.

G. I had been lobbying for built in array operations ever since I had returned from the University Of Michigan. He asked me to look into that, and not to forget complex numbers and double precision.

Looking back on this list, I can see that he was clearly thinking in terms of a network of special purpose processors for editing, I/O, array processing, etc., all communicating through something that he called *Sesame* control. I think that the term *Sesame* was intended to imply that if a process needed something from another process, then all it would have to do is say “Open Sesame”, and a magic door would open and there it would be.

To solve the up-level addressing problem, he described a set of display registers to me which were indexed by the lexic level of the procedure. This solution is clearly on the path that led to the B6500 display registers. I do not know if the idea originated with Barton, but I suspect that it may have come from somewhere else (perhaps Randall and Russell). There was also some detail left out having to do with the way that the display gets updated when a procedure calls a formal procedure where the actual procedure passed is declared at a higher lexic level.

I undertook the task of solving the string manipulation problem since I knew the Character Mode and Stream Procedures better than anyone else. I launched off on some esoteric set theory study where I assigned arbitrary symbols to arbitrary sets and operated on them with union and intersection operations. It also occurred to me that if the language could describe itself (i.e. if the language and metalanguage were one and the same), and if the compiler were written in its own language, then all I would have to do to change the language would be to change, the metalinguistic description and recompile the compiler three times and I would have a new compiler for the new language.

While I was sitting around tying my brain in knots with thoughts such as these, Barton (or someone) felt that they had enough, so the days of consulting came to an end. It was a very mind expanding time for me, and I hope that it was useful for Bob and Burroughs. I always thought that I should keep this work, and the fact that we were almost certainly getting paid twice by Burroughs, a deep dark secret. A little later on I was surprised to find out that Brad MacKenzie knew about it, but we never discussed it.

One reason that someone up there (in Burroughs management) might have approved this arrangement was that the engineers were getting paid overtime because they were part of the Pasadena Plant Structure, but Automatic Programmers were still part of the Business Machines Group and we were above that sort of financial remuneration. So maybe someone took pity or us. Anyway, it made for a more pleasant Christmas in 1961.

Selling The B5000 To Stanford And UTC

On February 19, 1962 I traveled with Norm Kreuder and Al Gerlach to Palo Alto. Norm was in charge of the engineering of the entire B5000. Al was in charge of the engineering for the B5000 processor. We flew in Norm's airplane. I received warnings from Lloyd, my secretary, Norm, and the Personnel Department that I would not be insured in case of an accident. I picked up Norm and Al at their homes in Glendora. We drove to Bracket Field and took off. The flight seemed to take forever.

Norm and Al were the first Burroughs engineers that, I had ever talked with. In those days, programmers and engineers never spoke directly with each other. All communication went through the Product Planning Department. Lloyd and I had waged two battles to get changes in the B5000. Lloyd wanted a new type of descriptor called the label descriptor. He won his battle. I wanted the elimination of the flagged data feature. I lost my battle. Paul King had told me that it was simply too late to get rid of the feature. Norm asked me what I would like to see changed on the B5000 and I told him that we could generate better code in some cases if it weren't for flagged data. Norm told me that if he had known that at the time that I was arguing with Paul King, that it would have been a trivial change.

We were met at the Palo Alto airport by Al Collins. I had heard stories about Al for years, but had never met him. When Al transferred from Dallas to Pasadena, it left the opening that I was hired to fill. The rule of thumb was that a district could have two Technical Representatives for each 220 plus one for every 205. Just before I moved to Pasadena, Al had moved on to Palo Alto run the Stanford computer center. Al was fired on the spot one day by Ed

McCollister when he read some decision that Burroughs had made and posted on the bulletin board outside of McCollister's office. Al found the decision particularly objectionable and said "What we need around here is a good corporate airplane crash".

There were three purposes for the trip:

A. I was going to make the first of two talks at a computing seminar at Stanford.

B. The local salesman, whose name was Joe Hootman, wanted Norm and Al Gerlach and me to help him sell a B5000 to Stanford.

C. Joe was also close to selling a B5000 to United Technology Corporation (UTC), but he had told them that we would have a Fortran compiler, which was not true. So I was supposed to tell them why that did not matter. We were going to have a Fortran to Algol translator. So I was going to use that to soften the blow.

The Burroughs salesman took us to Stanford and introduced us to Dr. George Forsythe, who was head of the Stanford Computer Sciences Department, and a Dr. Harriot, who seemed to be second in command. Dr. Forsythe was a very gentle man and made me feel quite comfortable. Norm did most of the talking. I answered all of the questions about the software. I remember Norm telling Dr. Forsythe that if he were going to teach a course on railroad trains, that he shouldn't be teaching the use of steam engines; he should be teaching about diesels.

The following morning, Joe picked me up at my motel room. He had Rick Truit, a Sales Technical Representative with him. We drove to UTC. I was to meet Dr. Gene Thompson, who was head to the UTC computer department. His office was in a converted house trailer. He wasn't there when we got there, so I had a few minutes to took around. I noticed that in his in-basket was an unopened ACM Journal that had the most comprehensive description of Algol 60 that had been published to date. It was entitled the "Structure And Use Of Algol 60" by H. Bottenbruck.

Dr. Thompson had some very odd mannerisms. When he entered his office, he did not wait for introductions. He looked and me and said "I do not know you. I don't like the way that Burroughs continually runs a parade of people through my office. Why are you here?". I wasn't ready for that onslaught. I thought to myself "Just who in the hell are you to be talking me like that. After all I'm the inventor of the Algol Syntax Chart, Recursive Descent, and I've faced practically all of the great minds of the computer industry, and came off quite well thank you; and I'm here as a special guest to make speeches at Stanford University". I suspect that my head was somewhat larger than even Dr. Gene Thompson's. All of my little soothing speeches went out of the window. I said "I am here to tell you that there is not going to be a Fortran compiler on the B5000". Then he said "Then I can tell you that there is not going to be a B5000 at UTC". I said "Anyone would be insane to use Fortran when Algol 60 is available to them". It seemed that everyone else was afraid to talk to Dr. Thompson like that. He seemed to appreciate my straightforward approach.

Then Thompson said "Algol is too difficult to learn. There is no good documentation on it". I reached into his in-basket and ripped open the envelope containing the Algol document and plopped it in front of him. I said "Maybe you should read your mail sometime". That completely destroyed him. He asked if I had time to talk to some of his programmers. I told him that I didn't because I was due to make a talk at Stanford in a few minutes. But that I was coming back to make my second talk at Stanford on March 7, 1962 and I would allow some time on that trip. He said that he would really appreciate it if I would do that.

I was then taken to Stanford and sat it the front row of the class. Dr. Forsythe made a few remarks and then introduced me to the class. As I turned to address the seminar, I saw that there were about 50 people present, including one Dr. Gene Thompson. We smiled at each other and I knew the sale of one B5000 was in the bag.

My first talk lasted 90 minutes. Everyone was curious about this strange machine. After the talk, Al Collins, Al Gerlach, Norm and myself went to the airport and they gave me a guided tour of San Francisco from the air. Then we returned to to airport and got in Al Collin's car and they gave me a tour of San Francisco from the ground. We went to the Hungry I night club, where Jack E. Leanord was playing. Norm sounds exactly like Jack E. Leanord and therefore was a big fan of his.

On the following day we flew back to Bracket field. The flight back was along the sea coast and took four and a half hours. I had recently driven it in my Corvette in six hours. I thought we would never get there.

A few days later Al Gerlach had worked late at the office. As he was driving home he fell asleep at the wheel and hit the center divider at the intersection of Huntington Drive and Irwindale Road. His car rolled over and he was thrown out and killed.

My second trip to Stanford was on a commercial jet, so it was much more comfortable. I spent the entire morning at UTC talking with Dr. Thompson and his programmers. The order for the B5000 was signed shortly thereafter. Dr. Thompson continued to call on me whenever problems arose for the next several years. He was also in the audience for my second talk at Stanford.

My second talk lasted one hour. I then went to the computer sciences building (which was an ancient old stone building). On the third floor was a large room with some beds in it for the computer center employees. Several of us sat around on these beds and talked about Algol and the B5000 well into the evening.

It was becoming clear to me that Dr. Forsythe and Dr. Harroit wanted a B5000. But the best that Burroughs would offer was about a 40% discount. IBM was offering them the world to take an IBM 7090. Finally, IBM said that they would give Stanford the 7090 for free plus a gift of $900,000 that could be used to build a new computer center. Stanford had a Burroughs 220 and IBM was determined to dislodge us. Stanford accepted the offer and used $400,000 to build the computer center and $500,000 to buy a B5000. So IBM had unwittingly paid Stanford to buy a B5000, much to their dismay.

There was much joy in Pasadena when we heard the news. The story was quickly relayed to Bob Forrest. Bob was a friend and neighbor of mine, and a former Burroughs employee. He is the best technical writer that I have ever had the pleasure to work with. He was currently the editor of Datamation. He loved the story of how Stanford had slickered IBM, so he published it in the following issue of Datamation and Burroughs received some free national publicity.

After the death of Dr. Forsythe, Professor John McCarthy became much more prominent at Stanford. McCarthy designed a list processing language called LISP. His implementation relied heavily on his management scheme for the secondary memory. It was a real pig on the B5000. He came to Pasadena to find out how to defeat our virtual memory system. No one was very interested in helping him. Dave Dahm spent some time with him.

The students at Stanford loved the B5000. It sat next to the 7090 and could run circles around the 7090 for student jobs. The turnaround time was terrific. But McCarthy could not stand the machine. So he imposed a minimum turnaround time of three hours on any work that was done on the B5000. The machine's popularity quickly fell and it was replaced by an IBM 360.

McCarthy came to Santa Barbara in February 1979. He delivered a paper to the local chapter of the ACM. He and I sat across the table from each other. I am sure that he did not recognize me. It had been 17 years since we had seen each other. After all of those years, I still felt uncomfortable with my feelings about circumstances relating to the loss of the Stanford B5000.

His wife had recently died while trying to climb K-11 with an all woman team. All of the ladies were from the United States except one. She was from England. John's wife and the English lady fell to their death. His wife wrote a letter before she left. It said, "If I die on this mission, I want my body to be left there". She did die. No one can find her body. I would have liked to have met that lady.

Bob Bock's Associative Memory

Lloyd Turner told me about a type of Gin that wasn't made with Juniper Berries. So people who were allergic to Juniper Berries could drink it without getting sick. I had never used Gin, and I had never heard of a Juniper Berry, but since Lloyd said this to me, I thought it must be a really good thing. The name of the Gin was *Oso Negro*, which means "Black Bear" in Mexican. With each bottle of Oso Negro that one buys, there is a keychain with a plastic black bear on it. The bigger the bottle, the bigger the bear. People used to walk around with a lot of black bears hanging from their keychains.

The problem with Oso Negro was that it cost a fortune in California. Someone told me about San Luis, Mexico. There was a California state law which prohibited people from bringing alcohol from Mexico into California, but San Luis was straight south of Yuma. Arizona. So the game was to drive to San Luis, buy whatever, cross into Arizona, and then drive back to Pasadena.

In San Luis, Oso Negro Gin and Vodka cost 98 cents per fifth. Barcardi Rum was 98 cents per fifth. Kaluha was eight dollars a fifth in Pasadena. In San Luis, it was $2.40. The law governing importation of alcoholic beverages into the United States was that one could bring back one gallon per month per citizen. A good approach to the law was to gather up all of the neighborhood kids and bring back a gallon per kid. Ken Meyers tried it one August afternoon in his black Oldsmobile convertible. Unfortunately, black isn't the best color in 130 degree weather. And that day, it was 130 degrees in San Luis. Ken actually had a gallon of Vodka burst in the trunk of his car.

I told Bob Barton about this wonderful place and he was anxious to go. So we climbed into his 1958 Chevrolet and headed for Mexico. It was during the winter of 1962 as I recall.

Bob had been on some kind of mental trip whereby he had decided that computers were no good. He would challenge me with questions like "Tell me one good thing that computers have done for society?". But this day was different: He was preparing for a trip to Paoli. He had to address some kind of technical seminar. He had me for his captive audience to try out his ideas. We were in that car for about 18 hours and he never stopped talking about computers. I was exhausted when we finally got home.

Bob had three new ideas that he was presenting. The idea that was most impressive to me is now called Cache Memory. Bob was calling it an Associative Memory. His presentation was such a success in Paoli, that he was asked to repeat it in Pasadena. One of the Pasadena engineers really liked the idea. Unfortunately, I can't remember his name. He wrote a simulator and started some studies to find out how much could be gained by adding this kind of memory to the B5000. Before he reached any conclusions, he left Burroughs.

It had been several months between the trip to Mexico where Bob had first described the Associative Memory to me and his presentation to the Pasadena Plant. During that time, the same idea had been published by someone at General Electric. There were the usual charges by Bob's detractors that he had stolen the idea. Bob never seemed to be bothered by such charges. I was more bothered by them than he was.

After the young engineer who had written the simulator left Burroughs, no one did anything with the idea for over a year. Then another engineer named Bob Bock picked it up and wrote some kind of report on it. But he apparently did not fully understand the idea, because the scheme that he had for selecting what got pushed out of memory simply was not workable. At the conclusion of the Algol project, Brad had told me that he didn't care if I didn't do anything for three months. A little later he came to me with a project that I could study just to keep from getting bored. He and Norm Kreuder were calling it Bob Bock's Associative Memory". I couldn't believe that they didn't remember the origin of the idea, but I took every opportunity to straighten people out.

I reconstructed the original idea to make it workable again. My study showed the speedup that could be gained with two different sizes of memory. I thought my report made it clear that the idea should be developed into a products but it was never implemented on the B5000.

The Migration to Pasadena

Al Collins was the first of the Dallas people to move to Pasadena, which I considered to be the home office. Bob Barton was at Shell Development in Houston. His first employee was a crazy Frenchman named Gerard Guyod. Everyone was getting angry at Shell Development. Gerard was the first to leave. John Pale hired Gerard in Dallas. Gerard told John about Barton, and the unhappy situation at Shell. John got someone in Pasadena to contact him. Not only Bob moved to Pasadena, but he brought his entire group. That consisted of Joel Erdwin, Dave Dahm, George Logaman, and Clark Oliphint. They were called the arthropods (i.e. Shell fish). Gerard followed Barton to Pasadena after a few months. Then Bob convinced me and Loyd to come to work on the Algol compiler.

My best friend in college was Jeff Landreth. Jeff and I played a lot of tennis. After graduation, Jeff went to work in Fort Worth. After my move to Dallas, Jeff and I started playing tennis again. One day I said, "Jeff driving 32 miles every time that we want to play tennis is ridiculous. Why don't you try to get a job at Burroughs?". So I introduced Jeff to John Hale and John hired him. Jeff's roommate in Fort Worth was Dick Shobe. So Dick followed Jeff to Burroughs. Jeff worked two or three years for John without taking any vacation. There was an unwritten agreement between Jeff and John that he would take his vacation someday. Lloyd encouraged John to moved to Pasadena to take charge of the B5000 Cobol project. Jeff said that he followed John to Pasadena so that he wouldn't loose his vacation time.

Lloyd brought Bobby Creech and Dave Dahm from Texas. Dave's roommate at Rice was Cal Zethreus. So Dave encouraged Cal to come to Pasadena. John brought Dick Shobe to California after he had spent some more time in school. So that is the story of how Automatic Programming become so heavily populated with people who had moved from Texas.

Conclusion

The break even point for the B5000 was 35 machines. The first year we sold 17 machines. The second year we sold 15. The third year we only sold one machine. Everyone knew that the life of a computer product was three years. So after all of our blood, sweat, and tears, it looked as if we had produced a product that was not profitable.

Ray Macdonald was President of Burroughs by now, and he told us that the next machine would not be an architecture like the B5000. It was going to be a much more traditional machine that our salesmen could understand. The number one goal for the machine was that it would be profitable. It would be a Cobol machine. It would have Fortran and not Algol. It would even have an assembler. After all these years of convincing people that they should do all of their programming in high-level languages, we were going to write our operating system in assembly language.

I felt heartbroken. Then a couple of very significant things happened. IBM came out with the 360 and started delivering a sales pitch that was straight out of the Burroughs B5000 book. They started talking about multiprogramming and doing away with assembly language programming. They stopped pushing Fortran and started promoting a block-structured language with Algol-like procedures. It was called PL/I. Also our disk based operating [system] was completed. We reannounced the B5000 as the B5500. Many an IBM salesman would get someone turned on to all of these wonderful concepts, only to find that the only machine in the world that would perform them was a B5000. I am sure that IBM convinced more people of these concepts than Burroughs did.

So the rumors of the death of the B5000 were slightly exaggerated. I don't know exactly how many B5000s were sold, but someone told me when the decision was made to not stop production at 200, but to build another 100. So our story has a happy ending after all.

Post Script

It has been five months since the first edition of this book was distributed. I only intended it for a few people in adjacent offices in my hallway. It occurred to me that Warren Taylor would probably enjoy it, so I sent him a copy. His secretary laid it in his in-basket while he was out of the office. Before Warren returned, a contingent of Detroit visitors came into his offing and saw it. Warren said that all hell bloke loose. People were tearing it apart and passing the different chapters around and saying “Listen to this”.

Copies were made and taken to Detroit. Both John Hale and Dick Shobe received calls from Detroit asking them to ask me not to publish it. Lloyd Turner was promoted to Vice President of Corporate Engineering. A few days later he came by and told me that he had read every word of it. He had one correction which has been included in this edition. But he also asked me not to publish it

I have received phone calls from Northern California and from the East Coast. One person offered to pay my expenses if I would come to Santa Clara and make a talk. Larry Wheelright took a copy to Manilla. Except for Detroit, all of the comments have been very favorable.

Some of the stories that I heard second-hand may not be accurate but this is the way that I heard them. The stories are primarily intended to reflect the intense attitude of the people involved. If anyone knows of any inaccuracies, I would be happy to hear about them.

I would offer to correct the inaccuracies, but tomorrow is my last day at Burroughs. I doubt that I will transfer this file to my new job, which is with Sperry Univac Mini-Computer Operations in Irvine California[12]. Sperry offers a unique opportunity for rapid advancement. I hope that at it will be similar to the period described in this book. But I will surely miss Burroughs.

Ian Joyner's Notes

1 I don't want to interrupt the flow and eloquence of Waychoff's article at all, but only offer these notes to help those unfamiliar with Burroughs equipment and to bring some issues up to date (at least as far as I know). I originally transcribed this paper into Word in the 1990s for internal Unisys distribution, but having just found the original again take this opportunity to transcribe it into OS X Pages and PDF, and make this entertaining story available to the world. I have made some small editorial changes like put apostrophes before s.

2 In 2007, this architecture has now survived a remarkable 43 years, now as Unisys Clearpath Libra MCP systems.

3 Of course, now multi-pass compilers are the norm for more sophisticated languages in order to avoid forward references and provide more sophisticated semantics checking. However, a one-pass compiler was indeed an achievement at the time, since multi-pass compilers meant tapes had to be rewound or card decks reloaded.

4 I met Dave Dahm once. He invented an extremely efficient locking mechanism which could avoid a system call in the case where a resource was not locked by any other process, using the readlock mechanism. Amusingly, this was implemented using defines! The mechanism was known as Dahm's locks.

5 SDL was the system language of the B1700.

6 Interestingly, the later B1700 SDL environment separated the stacks into at least a control stack and a data stack. However, this was a microcode implementation which would not have required any hardware, where perhaps the B5000 would have required extra hardware and internal registers.

7 This device sounds like the 60s and 70s Mellotron musical instrument.

8 Roy Guck was the inventor of the B6700 system library, I believe, which is the nicest and most powerful library ever invented, providing common access to shared resources. Today, these would probably be called *components*. His son Randy was also a genius working in the Mission Viejo database section before eventually leaving Unisys altogether. I met Randy several times. Apparently Roy was very unassuming and modest about his accomplishments.

9 Indeed line numbers outlived tape and card equipment and proved a very effective source control system, where every line was effectively assigned a unique id (UUID of sorts). Any patch could be undone and redone out of order, and the technique of being able to patch from separate files with the compiler still survives to great effect.

10 SPO was Supervisory Printer Output, the system console in other words, BOJ and EOJ were Beginning of Job and End of Job respectively.

11 Clear/Start is the Burroughs system boot procedure since you pressed the clear button and then the start button.

12 Burroughs had to buy out Sperry Univac in 1986 to become Unisys in order to get Richard Waychoff back.