Colossus and the Origins of Programmability

Draft for discussion at the 2016 SIGCIS Workshop.
Do not quote or cite without permission of the authors.

Thomas Haigh (thomas.haigh@gmail.com) and Mark Priestley (m.priestley@gmail.com)

This is an initial draft of some material from our ongoing project to explore the history of Tommy Flowers, the ways in which Colossus was used and configured, and its place in the history of information technology. This work is sponsored by Mrs. L.D. Rope's Second Charitable Trust. The current draft is very preliminary, so please do not quote, cite, or distribute without our permission. If names and references are confusing to you our apologies! Fortunately the basics of Colossus and Bletchley Park are well covered in Wikipedia.

Colossus was a codebreaking device built by the British General Post Office at the end of 1943, under the direction of career telecommunications engineer Tommy Flowers. It entered use at Bletchley Park in 1944 to speed the work of codebreakers targeting what was then codenamed "fish," a family of Lorenz teleprinter codes used for high level German military communication. Colossus was not a one-off machine, like most early electronic computers, but the prototype for a family of ten "colossi," in which later models incorporated some significant improvements. The machines were used by the Newmanry, a group under mathematician Max Newman, where ingenious codebreaking users discovered new applications for them which, in turn, shaped the provision of additional controls and capabilities in the later models. Unlike ENIAC, which began in modest secrecy but soon graced the front page of the New York Times, Colossus was highly classified and remained unknown to the public until the 1970s.

Although famous, its place within the history of computing remains ambiguous and its basic capabilities are little understood. In particular, Colossus is often said to have been a "programmable electronic computer," indeed to have been the first such computer, but closer attention shows that "programmable" has never been properly defined in this context, being taken to mean no more that "extensively configurable." Instead we pull together evidence form other mid-1940s projects, and from the use of "program" in other contexts, to argue that to follow a program is to carry out a sequence of operations over time. In this sense Colossus carried out a program, but although many parameters could be set for some of these operations the basic sequence of operations could not be altered by the user.

This characterization fits with statements made by Flowers himself, and some of those who worked with him. Their more nuanced characterizations, for example of Colossus as an "electronic processor" rather than as a "computer," or as a machine that followed a program set by Flowers rather than by the user, are notably more restrained than those of the machine's boosters. We argue that Colossus was shoehorned into the role of programmable computer at a time when it might have seemed that only this would restore its place in history. Today, as out reliance on digital communications grows while the traditional artifact of "the computer" vanishes we are better able to appreciate this remarkable machine on its own terms. As a pioneering digital signal processing device, Colossus was essentially unique among many wartime computing and codebreaking projects in making aggressive use of digital electronics, working reliably, being ready in time to help the war effort, and serving a vital role within the conflict.

Situating Colossus

Dozens of unique electronic and mechanical computers were built during the 1940s. Of these a handful, such as the Harvard Mark 1, ENIAC, EDSAC, and the UNIVAC 1, have clear and prominent places in the history of computing. They consistently appear in overview histories, such as Martin Campbell-Kelly and Willian Aspray's *Computer* and Walter Isaacson's *The Innovators*, in television documentary series, and in comprehensive museum exhibitions such as those at the Computer History Museum and the Heinz Nixdorf Museums Forum. Each is remembered as the "first" machine to reach one or another historical milestone. These specific honors were agreed upon after a long and messy battle, conducted during the 1970s and 80s, over which deserved to be called the "first computer."

The historical place of Colossus is less clear. Most other pioneering computers were publicized during their operational lifetimes. ENIAC, for example, was announced to the world with a front page story in the *New York Times* and installed in a showpiece facility where it was frequently displayed for visitors.[1] The Colossus machines were designed in secret, deployed as a vital part of one of the war's most militarily sensitive operations, and kept confidential for decades afterwards. From the 1940s to the 1970s, as teams of patent lawyers gathered records concerning other early machines and subjected their designers to repeated rounds of deposition and testimony, those responsible for Colossus remained quiet about the machine's capabilities and even its existence.

Word of Colossus began to spread in the 1970s as Brian Randell, a computer scientist with an interest in the early history of electronic computing, heard rumors about the machine's existence. In 1976 he shocked the computer pioneers a seminal computer history meeting at Los Alamos National Laboratory with a detailed account of Colossus and the conclusion that Colossus was "a special-purpose program-controlled electronic digital computer" that could "most aptly be compared" to ENIAC in its flexibility and programming method.[2] Randell had dug up an impressive amount of information, but without access to original documents his account was unavoidably speculative. This led to considerable interest in Tommy Flowers, who gave several public talks and interviews in the 1970s and published his own technical article on Colossus and its history in 1983. Yet even Flowers was working from memory, and his description turned out to have several significant historical and technical inaccuracies.[3]

This factual uncertainty as to what Colossus actually was and how it actually worked may explain its strange position within the history of computing. It is celebrated by a small community of enthusiasts, some of whom view it as the most important of all the early computers. For example, Jack Copeland has claimed that if the Colossus machines had been preserved as "the heart of a scientific research facility" then "the Internet—and even the personal computer—might have been developed a decade or more earlier."[4]As with boosters of other less well known early computers, such as the proponents of Iowan

[1] Thomas Haigh, Mark Priestley, and Crispin Rope, *Eniac in Action: Making and Remaking the Modern Computer* (Cambridge, MA: MIT Press, 2016),

[2] A revised version was published the seminal volume Brian Randell, "The Colossus," ed. N Metropolis, J Howlett, and Gian-Carlo Rota (New York: Academic Press, 1980)

[3] Thomas H Flowers, "The Design of Colossus," *Annals of the History of Computing* 5, no. 3 (1983) For example, Flowers remembered the electronic buffer introduced on Colossus 2 as buffering the outputs of five channels of simulated cipher wheel output, but later analysis showed that it buffered only one channel.

[4] Copeland, p. 119 of Turing popular book.

computing pioneer John Atanasoff, this discourse can take on a rather partisan tone.[5] Copeland, for example, attributes to Flowers himself remarks that "Colossus was 'much more of a computer than ENIAC'" which was "just a 'number cruncher.'"[6] Tony Sale, who devoted many years of his retirement to the reconstruction of Colossus, wrote that his project to rebuild Colossus has been motivated in part by a sense that "for far too long the Americans have got away with the myth that the ENIAC was the first large-scale electronic digital calculator in the world." Following the completion of the reconstruction, he claimed, "There has been a stunned silence from across the water."[7]

Within Britain, this Colossus was recently honored with a stamp from the Royal Mail, on which was printed "world's first electronic digital computer," and at least in Britain retains a fairly high profile thanks to its connection with the work of Bletchley Park, which has now eclipsed former favorites such as the "dambusters" raid to become one of the most famous and celebrated aspects of the war. Colossus even makes a brief appearance in *Cryptonomicon*, Neal Stephenson's hugely popular novel of cryptography and the wartime origins of information technology.

It is far from clear that the more recent silence noted by Sale is the result of stunned acquiescence by "the Americans" to his claims. Colossus seems rather to have been politely ignored by most serious historians of computing. In *Computer*, Campbell-Kelly & Aspray note merely that Turing's design for a mechanical codebreaking machine "was followed by an electronic machine, the Colossus, in 1943," as a result of which several people who would later work on computer projects were exposed to electronic technologies.[8] The other standard scholarly history of computing, Paul Ceruzzi's *A History of Modern Computing* doesn't mention Colossus at all.[9] Perhaps as a response to growing awareness of Colossus, Ceruzzi gives it about a page in his shorter and more recent *Computing: A Concise History*. He wonders why Colossus is not "more heralded," and concludes that its focus on textual rather than numerical operations had combined with the long prevalent secrecy to marginalize it.

The current situation, then, is one in which Colossus is lavishly, if sometimes shrilly, praised by its fans and increasingly embraced by the British public as a symbol of national greatness but largely ignored by

---

[5] For example, Alice Burks, *Who Invented the Computer: The Legal Battle That Changed Computing* (New York, NY: Prometheus Books, 2003),

[6] "Colossus and the Rise of the Modern Computer," in *Colossus: The Secrets of Bletchley Park's Codebreaking Computers*, ed. Jack Copeland (New York: Oxford University Press, 2006)

[7] http://www.codesandciphers.org.uk/lorenz/rebuild.htm

[8] Martin Campbell-Kelly and William Aspray, *Computer: A History of the Information Machine* (New York, NY: Basic Books, 1996), 99-100. The more recent third edition retains the same text (p.82).

[9] Paul E Ceruzzi, *A History of Modern Computing* (Cambridge, Mass.: MIT Press, 1998), .

historians shaping broader narratives on the emergency of modern computing. This is in part because of an enduring vagueness surrounding the question of what Colossus actually did. Randell's early characterization of the machine as both "special purpose" and "programmable" was based on fragmentary evidence and oral testimony from veterans of the project. To the best of our knowledge, neither Randell nor any other scholar has attempted to define specifically what "programmable" means in this context.

This characterization has endured; for example the Wikipedia page on Colossus currently describes it as "the world's first programmable, electronic, digital computer." In his recent history of the relationship between computers and cryptography, Paul Gannon reshuffles the adjectives slightly: "Colossus can be defined as an, (sic.) electronic, binary/logic-processing, programmable, specific-purpose machine."[10]

Yet it is far from clear what "programmable" means in this context. The usual distinction is between computers that are "general purpose" and can be reprogrammed to do different kinds of tasks, versus those that are "special purpose" and run a particular program. For example, the embedded microprocessors in DVD players, cash dispensers, or airbag units are the modern descendants of early special purpose computers built for tasks such as missile guidance or toll collection. Colossus is, to our knowledge, unique in being widely characterized as both special purpose and programmable. While a great deal of new information has become available about the specific functioning and use of Colossus, thanks to declassification of archival records and the work done by the rebuild team, no systematic effort has been made to reevaluate the position of Colossus as a programmable special purpose computer.

Was Colossus a Computer?

Since its rediscovery in the 1970s, boosters of Colossus have no doubt that it was a computer, indeed that it was the first and most important electronic computer. Given this, we were startled to notice that Flowers himself remained reluctant to call Colossus a computer, preferring the different and in our view more interesting characterization of Colossus as an "electronic processor."

Only many years after the Colossus machines were shut down did people begin to call them computers -- we certainly found no archival evidence that Colossus was ever called a computer during its operational lifespan. This sets Colossus apart most of the pioneering electronic computers of the 1940s, which were usually named either as computers (the "C" in ENIAC stood for "Computer" as did the "C" in EDVAC) or calculators (the "C" in machines such as EDSAC and IBM's SSEC). This is because the new machines replaced the labor of humans, whose job title was "computers."[11] They were often known as "automatic computers," just as machinery that could direct the flying of a plane was called the "automatic pilot" because it carried out some of the tasks of a human pilot.

Computers carried out lengthy mathematical tasks, often involving thousands of individual mathematical operations. This drew attention to their ability to move from one operation to the next

[10] Paul Gannon, *Colossus: Bletchley Park's Greatest Secret* (London, UK: Atlantic Books, 2006), P.433?

[11] While we know of no evidence that Colossus was called a "computer," some British codebreakers were known as "Computor Clerks," or sometimes simply as "computors." According to Michael Smith, "This curious title had nothing to do with electronic computers… but was an echo of an old War Office covername for cryptanalysts – Signal Computor." Michael Smith, *The Secrets of Station X: How the Bletchley Park Codebreakers Helped Win the War* (Biteback Publishing, 2011),

without human intervention. George Stibitz built a series of pioneering tape-controlled computers at Bell Labs during the 1940s. His 1945 definition captures the contemporary understanding of a computer as something able to perform automatically a sequence of operations ("some or any of" multiplications, divisions, additions, and subtractions) storing the intermediate results from earlier operations so they could be further manipulated by later ones.[12] More complex operations, such as square roots, logarithms, and trig functions, were handled in some early machines with special hardware and in others by specifying the appropriate string of elementary operations. Many of these machines were designed with table making in mind, so that the machine would crank out results for one set of parameters after another by constantly repeating the same processes. Colossus, unlike these other machines, was not built to carry out numerical computations and could not use the results of one step as input for a subsequent step. [13]

There is a popular misconception that Flowers was keen to work with computers after the war but was somehow unfairly thwarted. One anecdote, currently featured on his Wikipedia page, states that "Flowers applied for a loan from the Bank of England to build another machine like Colossus but was denied the loan because the bank did not believe that such a machine could work. He could not argue that he had already designed and built many of these machines because his work on Colossus was covered by the Official Secrets Act." This is implausible for several reasons, one being that as a central bank the Bank of England does not make loans to entrepreneurs. But, as borrowing money to build a privately codebreaking machine would not make much sense, it reflects an assumption that Flowers was keen to build general purpose computers.

In reality Flowers did not see Colossus as a computer and was never particularly interested in computing. There is little reason to doubt that he would have been able to find employment on a computing project after the war if he had desired this. Others who worked on classified electronic projects during the war played key roles in late-1940s British computer projects, including Alan Turing, Max Newman, Freddie Williams, and Tom Kilburn. Many years later he did complain that lacking "administrative or executive powers" he was unable to convince others to let him build on his wartime achievements, unlike others such as Turing and Newman who found "positions where they could use their knowledge effectively without disclosing the source." But he again made clear that his desire had been to revolutionize telecommunications, not computing: "I was in a similar position in the telephone industry—except for having no power or opportunity to use the knowledge effectively."[14]

In fact Flowers had an opportunity to work with Turing and Newman in 1946 when Dollis Hill agreed to build a computer, ACE, for the National Physical Laboratory according to Alan Turing's design. Charles

[12] Stibitz, 1945 AMP report "Relay Computers." ML27-b3 in his papers at Dartmouth College.p.2

[13] Jack Good, a veteran of Colossus practice at Bletchley Park, later claimed that, if appropriately configured, Colossus could almost have carried out a multiplication but that this would not have been possible in practice because of constraints on what could be accomplished in a processing cycle. We have no reason to doubt this, though it would presumably have required special settings of the code wheels and message tape and been, even if possible, a rather inefficient alternative to a desktop calculator. This fact has been offered as proof of the flexibility of Colossus, which in a sense it does attest to: a device designed without any attention to numerical computations could almost have multiplied thanks to the flexibility with which logical conditions could be combined. Yet it also proves the very real differences between Colossus and devices designed for scientific computation. Multiplications were vital to computations, and a device that could not multiply would not, by the standard of the 1940s, be termed a "computer "or "calculator."

[14] Cite D-Day chapter in Copeland compendium.

Darwin, the laboratory's head, wrote to the Post Office's Engineer in Chief promising tens of thousands of pounds for the project. Secrecy was no problem: "it works using principles developed by your staff during the war for a certain Foreign Office project, and we want to be able to take advantage of this, enlisting the help of your Research Department, and in particular of Mr. Flowers who has much experience in working out the electronic side of it."[15] NPL also approached BTM and TRE, the other wartime machine building partners of Bletchley Park, for help with the project.

Yet Dollis Hill let this work slip behind other projects, assigning it nothing like the priority it had given Colossus. When it withdrew from the contract it had carried out only six hundred pounds of billable work, a small fraction of what had been planned.[16] It is not clear from surviving sources whether Flowers himself was instrumental in making the decision to prioritize other projects, but there is certainly no evidence that he felt anything like the same passion for ACE that he did for Colossus. His subsequent career trajectory confirms that electronic telephone exchanges, not computers, were Flowers' driving force. In his talk, he complained that within exchanges electronics had most prominently, as of 1977, been used by Bell Labs to have a general purpose computer control the switches connecting together analog telephone lines, which he termed the "structure" of the exchange. He told the audience that he had spent his career urging that electronic "processors" be used to replace the analog switches, rather than control them, so that the "structure" of the exchange would itself become electronic. Flowers noted the "irony of recent events which credit me with some pioneering work on computers, when it was my refusal to use a computer when everyone else said it was the right thing to do that which led to my downfall in the telephone industry." This ultimately led to his departure from the Post Office when it seemed that this would give him an opportunity to develop his exchange technology elsewhere – a conspicuous contrast with his decision to remain with the Post Office after work stalled on ACE.

When, in 1977, Flowers gave one of his first public talks about Colossus, after news of the machine had begun to reach the public, he related that "it is now said that during the Second World War I was responsible for the production of the world's first electronic digital computer," yet cautioned that "if so, that was an accident incidental to the solution of a problem." [17] He himself preferred to situate Colossus within the history of telecommunications processing, saying that the challenge was "to break through to a new switching technique which we called and is now universally known as electronic, but no thought of computers was in our minds." Instead Flowers preferred the broader concept of a "data processor," arguing that claims for Colossus as the first computer came "to the surprise of those concerned who thought of it as just another processor." Telephone exchanges were themselves processors, in this sense, and Flowers spend most of his talk discussing them.

He cast Colossus as the first of these electronic digital processors, seeing it as extension of his work on electronic telephone exchanges. Going back to the years before the outbreak of war, he related that "under construction at Dollis Hill… was a processor using thermionic valves for high speed processing with maximum reliability, the processor being time shared among numerous smaller processors using valves relays [check] for simple and slow operations…"

The closest he came to claiming Colossus as a computer seems to have been this 1983 passage: "Colossus had features now associated with digital computers—semi-permanent and temporary data

[15] Dawin to Angwin, 22 Feb 1946, NPL PDF
[16] Another citation in the NPL PDF. There are several relevant documents in Copeland's online archive.
[17] 1980 NPL publication

storage, arithmetic and logic units including branching logic, and variable programming—that may justify its being regarded as the first digital computer." In the rest of that paper Flowers consistently calls Colossus a "machine" rather than a "computer."[18]

Decades after others won him recognition as an inventor of the computer, Flowers himself conspicuously avoided calling Colossus a computer. His posthumously published chapter "Colossus" opens with the sentence "Machines such as counters, computers, and Colossus process information." That choice of words suggests that he viewed Colossus as akin to both counters and computers, but not as itself a computer. In the rest of the article he uses the word "machine" rather than "computer" when talking about Colossus.[19] In another posthumous chapter, "D-Day at Bletchley Park," Flowers consistently favored phrases such as "electronic machine" and "processor" to describe Colossus. In that chapter, Flowers again attributed to others the idea that Colossus was a computer rather than claim this himself.

> [A]cademics interested in the history of computing have recognized that Colossus was the world's first electronic computer. It was not designed as a computer: computers had not yet been invented. It resembled a modern computer about as much as George Stephenson's Rocket locomotive of 1829 resembled the Royal Scot and other steam locomotives of the twentieth century. The basic technology used in a modern computer—data storage and retrieval, ultrafast processing, variable programming, the printing out of the results of the processing, and so forth—were [sic.] all anticipated by Colossus, some of it by as much as ten years.

Note that his assertion is "computers had not yet been invented," not "I invented the computer." What Flowers does claim credit for is the development of many of the key digital electronic techniques later used to build computers.[20] We should perhaps take him more seriously in this respect.

What Is a Program?

The concepts of program and programmability have received surprisingly little systematic historical examination.[21] The term "program" does not seem to have been applied to automation before the ENIAC project of the mid-1940s. Its initial use here mirrored the broader sense of the word, such as a concert program, program of lectures, or program of study. In each case the program is a description of a number of actions to be performed in a particular order. The concert program, for example, specifies a

---

[18] Flowers, "The Design of Colossus," P.252. His assertion that Colossus incorporated "branching logic" is questionable, and the meaning of "variable programming" is unclear. He also asserts that "Colossus was comparable in conception and processing power with the ENIAC" which reflects an understandable lack of awareness of ENAIC.

[19] "Colossus and the Rise of the Modern Computer," in *Colossus: The Secrets of Bletchley Park's Codebreaking Computers*, ed. Jack Copeland (New York: Oxford University Press, 2006)

[20] While we could find no article or talk by Flowers in which he calls Colossus a computer, Jack Copeland has made prominent use of quotations he attributes to unpublished oral histories with Flowers in which Flowers is less ambivalent.

[21] David Alan Grier, "The Eniac, the Verb "to Program" and the Emergence of Digital Computers," *IEEE Annals of the History of Computing* 18, no. 1 (1996) "Programming and Planning," *IEEE Annals of the History of Computing* 33, no. 1 (2011) We share Grier's general sense that the idea of "programming" was introduced to computing in the ENIAC project, but are not convinced by his suggestions adoption of the term reflected a desire to "establish the economic importance of automatic computing" or that the alternative term "planning" reflected a specific commitment to Taylorist conceptions of production engineering.

number of pieces of music to be performed by an orchestra on a particular evening. In each of these cases the programmer determines which of a set of possible actions should be performed and in what sequence they should be carried out. A television network programmer is responsible for choosing and sequencing shows to produce a schedule.

Automatic computing machines could switch from one mathematical operation to the next without human intervention. So in that sense every automatic digital computer was carrying out a program of operations, though that specific term was not always used. Most of the first generation of automatic computers, built during the early 1940s, relied on paper tape to control the operations performed, providing mechanisms for the unit reading the tape to direct the operation of the arithmetic units. This was not in the least novel. Player pianos and automatic looms had coded control information as holes punched in tape since the nineteenth century. Charles Babbage had proposed the use of a similar control mechanism for automatic computation when designing his never-built Analytical Engine.

Various terms were used to describe these automatic controls. Babbage followed standard mathematical terminology in calling the actions carried out by his planned engine "operations." He called the cards holding them "operation cards." The Harvard Mark I computer, built by IBM for Howard Aiken's Computing Laboratory, was controlled by a paper tape. The patterns punched onto the tape were called "codes," which appears to be the origin of "coding" in this context. The word "sequence" was often used to describe the content of a particular strip of paper tape, which usually represented either a loop (in which case the ends of the tape were joined to create a physical loop) or a subroutine. This ability to sequence and automatically perform operations was central to the new machines, something recognized in the titles IBM gave the Harvard Machine (the Automatic Sequence Controlled Calculator) and to its more ambitious successor (the Selective Sequence Electronic Calculator). Whereas the earlier machine had relied on human operators to change tapes once a loop terminated or a branch was reached, the SSEC had dozens of tape readers and could automatically shift control between them as needed, thus "selecting" the appropriate sequence to perform.

References to "programming" appear for the first time in the ENIAC project. ENIAC differed from the earlier computers in two important respects. Firstly, it used vacuum tubes rather than electromechanical relays for its arithmetic and memory circuits. These could switch thousands of times faster. Secondly, to exploit this speed, its designers fully automated its control. Rather than read control sequences from tape, its control circuits were driven by a network of wires carrying control pulses between different parts of the machine. The arrival of a control pulse triggered an action, the details of which were set using switches on the unit in question. The terms "programming" and "program" were not originally used to describe ENIAC's closest analogs to their modern senses: the act of configuring the machine to carry out a particular problem and the resulting configuration of wires and switches. Instead these were called, respectively, "setting up" ENIAC and a "set-up."

Instead, the word "programming" was enlisted to describe aspects of ENIAC's control mechanisms. As well as calling its control signals "program pulses," a June 1944 progress report described two ENIAC accumulator units as being "automatically programmed to receive the multiplier and multiplicand" when a program pulse triggered the multiplier unit to which they were attached. The adoption around the same time of "programmer" as the name for a simple mechanical control unit of an automatic washing machine reflects a similar usage – turning the dial to a particular point triggers the performance of a particular sequence of washing operations (spin, rinse, wash, and so on). Echoing this, a primary

meaning of "program" on the ENIAC was to describe a single operation set up on one of its units. What were being programmed were the operations of the internal circuitry of that unit.

By late 1945, however, the ENIAC team was beginning to talk of "programming" in something much closer to its modern meaning. The meaning of the verb "to program" quickly shifted from describing the action of the control circuits responsible for triggering operations at the correct time to describing the work of the humans devising such sequences. In late 1945 a letter from one of the project's leaders noted that "the EDVAC will contain a large number of units capable of remembering programming instructions, "to be copied from tape" before the actual program is started."[22] Similar terminology was applied to ENIAC; a report described the practices used in "planning a set-up for the ENIAC" as "programming techniques."[23]

The new meaning of program seems to have been connected to the new approach to automatic control formulated for EDVAC, the follow-on to ENIAC commissioned in the summer of 1944. John von Neumann's celebrated "First Draft of a Report on the EDVAC," circulated within the ENIAC team in April 1945, combined the established approach of controlling a computation by reading a sequence of coded instructions with the novel idea of storing these instructions in a large, addressable memory using the same mechanisms employed to store and manipulate data. While von Neumann called these instructions "code" and the process of producing them "coding" they were more commonly called "programs." The Moore School team adapted its existing use of "programming" to the new approach, altering its meaning in the process.

The adopting of "programming" in computing was a simple extension of its everyday meaning in other contexts. This is very similar to the meaning of "program" in other contexts – for example the work of a radio programmer who selects and schedules programs for broadcast, or the programming of a series of concerts. "Programming" an automatic computer involves the automatic execution of a series of operations over time. Likewise the other uses of the words "program" and "programming it involves a selecting not just which things should happen but also when they should happen. For example, a particular orchestra plays different pieces over the course of a concert. A particular lecture hall hosts different speakers over the course of a lecture series. A particular student takes different courses each semester over the course of a degree program. A washing machine fills with water, soaks, agitates, empties, rinses, and spins when the appropriate program is selected.

We therefore resist the idea of applying the idea of "programming" to configuration mechanisms that do not sequence different operations over time. For example, punched card tabulating machines could be configured to ignore some fields on each card they processed and to tally others to create totals and subtotals based on the data punched on a card deck. By the 1930s these configurations were rather complex. But the specified counts and evaluations were carried out in an identical way when each card was read. So we do not think that the concept of "programming" is applicable here since there was no succession of activities over time. In contrast, IBM's 604 Electronic Calculating Punch, introduced in 1948, could be set up (via a plug board) to perform a sequence of up to 40 steps each time a card was

[22] H. Goldstine to H. Curry, 3 October 1945, in the collection "ENIAC Patent Trial Collection" in the University of Pennsylvania archives.

[23] J Persper Eckert et al., *Description of the Eniac and Comments on Electronic Digital Machines. Amp Report 171.2r. Distributed by the Applied Mathematics Panel, National Defense Research Committee, November 30* (Philadelphia, PA: Moore School of Electrical Engineering, 1945), .

read. This would constitute a program, as the operations were performed sequentially rather than simultaneously, and in fact the earliest known use of the term "stored program" is to distinguish between a plug board program of this kind and one stored in an addressable computer memory.[24]

Likewise, in analog computers, such as differential analyzers, each part of the machine carried out the same operation throughout the course of the computation. There was no separation of data from control, and hence no sense in which one part of the machine was "programming" another to stop what it was doing and to start something different.

Some machines also allowed the mathematical units to direct each other. The Harvard Mark I, for example, included dedicated units to perform complex mathematical operations such as logarithm and sine functions. These were not self-sufficient, but instead directed the machine's other units to perform the necessary series of operations over the course of a minute or so. It seems reasonable to call this sequence of operations a "program" even though the precise term does not appear in contemporary sources.

So What Does "Programmable" Mean?

What about "programmable," the term applied by Randell and others to Colossus. Whereas "program" has a long history, "programmable" appeared only after the spread of the electronic computer. The Oxford English Dictionary shows no usage prior to 1953, in which year it documents the appearance of two distinct but related meanings: "Capable of being scheduled in accordance with a programme of events" and "Of an apparatus, operation, etc.: capable of being programmed." The OED assigns a still later date, 1964, to "programmability" which it defines as "the property of being programmable."

It's clear from this that the idea of "programmability" is significantly newer than the concepts of "program" or "programming" and that it is used primarily in the context of computerized control systems. Merely following a program is not enough to make something programmable. For example, an automatic washing machine incorporates a "programming unit" to control the sequence of operations it carries out.

Washing machine users can choose between several different programs by turning a control dial to the desired starting point, and on many models can also push buttons to set parameters such as "light load" which modify the operations performed. Yet the idea that a washing machine is "programmable" seems odd, and it is very unusual to talk about "programming" a washing machine.[25] The conventional term is that a user "selects" a program cycle.

[24] Thomas Haigh, Mark Priestley, and Crispin Rope, "Reconsidering the Stored Program Concept," *IEEE Annals of the History of Computing* 36, no. 1 (2014)

[25] Google currently finds 94 uses of the exact phrase "programmable washing machine" which suggests that it is rarely used.

On this typical washing machine dial, users select one of three predefined programs, such as the normal program which runs as follows: hot, warm, cold, rinse, spin, off. Users can also skip to a particular point within each program, for example starting the normal wash program at the beginning for heavily soiled clothes or omitting the hot wash operation for regular soilage.

Likewise, in the computing context one would not usually call the act of choosing a program and running it on a computer "programming." Such a usage might resonate with the original ENIAC sense of programming, in which directing the operation of a piece of machine was programming it, but the concept of "programmability" appeared only after our modern sense of a computer program was well established. Thus the idea "programmability" has historically been applied only to a specific, narrower and later sense of "program" and, as the OED points out, means that a user can establish a schedule of events.

Historically, there doesn't seem to have been much discussion of "programmability" in the 1960s there was little need to distinguish rigorously between programmable and non-programmable computers. Anything described as a computer was understood to be programmable. The term gained new currency in the 1970s following the introduction of powerful electronic calculators, where users could specify and store sequences of operations to be carried out automatically. Were these computers? The concept of a "programmable calculator" was introduced, with the understanding that it described a class of portable, personal machines that could be programmed by their users but which was more limited than true computers.[26] For example, a 1976 report "Calculators and the Computer Science Curriculum" reported that cheaper calculators are not "programmable by the user" even though "they do contain stored programs and can execute these programs" for example by pressing the square root key. Thus "there seems to be a clear difference between these calculators and what most computer scientists commonly think of as computers." In contrast, "programmable calculators" had "sufficient memory to store a series of key strokes (that is, a sequence of machine language instructions) and then to execute the program… At the higher price levels (but well under $1,000) such machines approach the ENIAC in

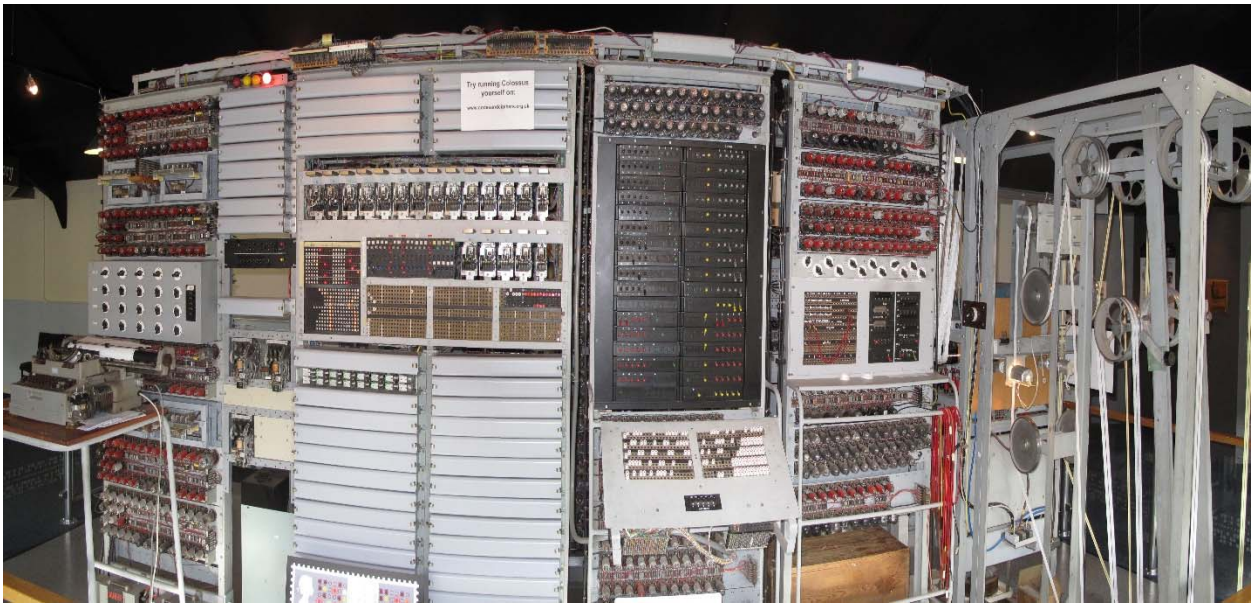
[26] This was more of a practical and marketing distinction than a theoretical one, as some programmable calculators had Turing complete programming languages.

capability, and will soon exceed it."[27] That was the same year in which Randell originally described Colossus as "programmable" and it seems reasonable to suppose that he had something similar in mind.

We conclude that the concept of "programmability" as applied to a computer requires not only that the computer carry out a sequence of distinct operations over time, i.e. to follow a program, but also that it allows its users to define new sequences of operations rather than just choosing between existing programs or supplying parameters.[28]

Was Colossus Programmable?

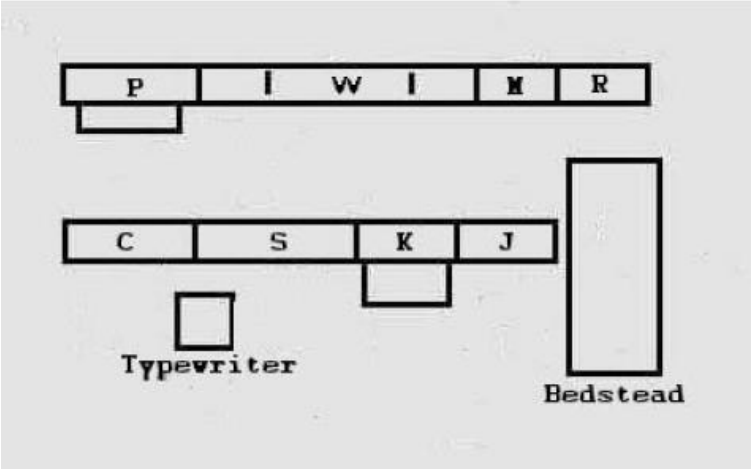
Colossus certainly carried out a program, in that the interaction of its control circuits with the message tape caused it to carry out different operations over time. A special code punched at the end of the message triggered control signals to reset its counters and, if a predefined threshold had been reached, to print the code wheel settings being evaluated and the counts obtained. Each message was followed by a blank sequence in the tape, which gave Colossus time to increment the uniservos holding the initial code wheel settings. These settings were then used to fix the positions of the electronic code wheels, so that when the tape spun round again to the special character that marked the beginning of the message the machine was ready to evaluate the message against the next possible combination of wheel settings.



The reconstructed Colossus 2 at the National Museum of Computing (Wikipedia photo)

27

28 Note, however that the OED also offers the definition for programmable "Of a control or facility: capable of being assigned a function by the user." This is presumably to capture the idea, for example, of a programmable function key or remote control unit in which the user is choosing a function but there is no obvious sequence involved.



Tony Sale's schematic – most controls are on rack C (far left) and racks K and J

Many years later, Harry Fensom, a senior member of the team that designed Colossus, reconstructed from memory the series of human and automatic actions it took to guide Colossus through a typical run. As he mentioned, "One panel of Colossus contained the so-called 'master control.' This acted as a program sequencer, guiding the run through all its steps, from switch-on, to print-out, and then on to the end of the run. Flowers designed the routine, or program, carried out by the master control, using a timing diagram and logic diagrams that had almost a modern flavour…"[29]

He documented thirteen manual actions to get the machine ready – loading a message tape, configuring the plug board with the appropriate logical inputs, setting the wheel start positions, and so on. These were followed by twenty automatic steps, such as resetting wheel positions, waiting for the message start signal, and comparing the counts to the thresholds selected by the operator. The sequence included inner and outer loops. The inner loop was followed each time a character was read from the message, the outer loop repeated each time the entire message had been read to reset the totals and increment the wheel start positions.

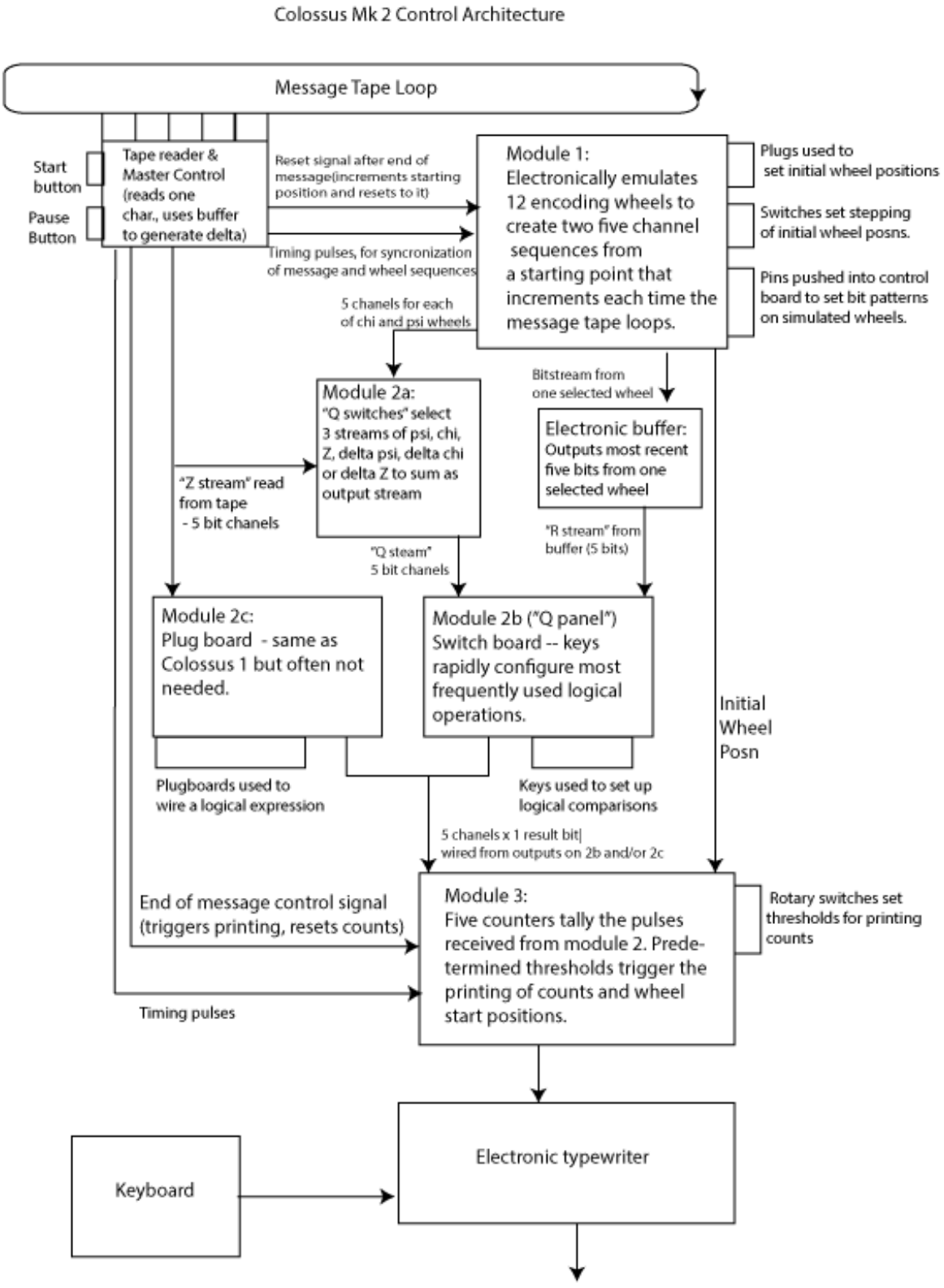
To characterize Colossus as "programmable," however, is to suggest not only that it followed a program but also that the program it ran could be changed without rebuilding its hardware. Notice that Fensom, quoted above, characterized Colossus as running a single program, designed by Flowers, rather than many different programs written by its users.[30] We think he was right, although this rests on a difficult question: what kinds of modification to the program executed by a machine are sufficient to make that machine "programmable?"

This is not a trivial issue. Colossus was highly configurable, and so could be used for many different tasks – a tribute to the foresight of Flowers' team and its close liaison with Newman and his group at Bletchley Park. Colossus was designed with wheel setting in mind, which began by taking the difference between two consecutive message bits on two of the five channels and comparing this with the signals coming from the corresponding Chi wheels. Once settings for the first two wheels were determined, two more

[29] P. 303 of Colossus compendium.

[30] Flowers himself, in 1983, described Colossus as following a "master control program," which was fixed by the control unit, but also called Colossus configurations set up with jacks and switches "programs" and the person configuring Colossus a "crypanalyst-programmer."

runs would identify the settings for the other three wheels. The codebreakers found many other ways to use Colossus for different parts of the coding job. For example, one way to determine whether decryption settings were correct was to count the number of times each letter appeared in a message. Colossus could be set to look only at the characters on a message tape (ignoring the simulated cipher wheels entirely) and to count the frequency with which each appeared. Unencrypted text is made of words and, as any Scrabble player knows, the distribution of letters in natural language words is highly irregular. In a well encrypted bitstream, however, all codes are equally likely.



This diagram abstracts away from the physical complexity of Colossus 2 to reveal the architecture of its control system, including the main user-configurable settings.

Settings configurable by the user included:

In logical "module 1," which emulated the Lorenz cypher wheels (physically Rack R for the bit settings, Rack S for the steppings)

- Plugs to set the initial wheel start positions
- Switches to control the stepping of these initial positions each time the message was restarted
- Pins to set the bit patterns on the simulated wheels (known as "triggers")
- (something) to determine which code wheel's output was buffered so that five wheel combinations could be evaluated simultaneously against the message

On Colossus 2 the corresponding physical units were found in Rack J (the selection panel, span counter, and jack field), Rack K (the Q panel for switching) and Rack W (the electronics for the simulated wheels). These correspond, with greater convenience and efficiency, to the options available in what was called the "combining unit" on Heath Robinson, a kind of prototype for Colossus. So we will use that phrase to combine their collective capabilities.

- Switches to determine which of the many possible inputs were fed through to the switch panel, and whether to feed raw values or "deltas" between successive bits
- Wires set on the plug board, to run selected inputs through particular combinations of logic gates
- Keys set on a switch board, to run selected inputs through particular combinations of logic gates
- Connections from particular logic gates to any of the five counters
- Span counters, to specify which sections of the message should be included in the count (Not shown on diagram)

In logical "module 3," which included the counters (Rack C) and printers

- Rotary switches to set the threshold above which the contents of a counter, and corresponding initial wheel positions, should be printed

The logic circuits in the "combing unit" were the most flexibly configurable part of Colossus, and underpin characterizations of Colossus as programmable. Plugboard cables and, on later models, switches used to select inputs and run them through logic gates to generate pulses for the counters. For example Benjamin Wells, who has looked in detail at Colossus, wrote that "tightly refined codebreaking algorithms were implemented in plug-wiring and switches. But the crucial story is that the same machine supported many different algorithms via flexible programming."[31]

Users had a great deal of flexibility in configuring its circuits to run inputs from the message tape and electronic code wheels though different logic gates to combine them in different ways for different purposes. Signals from the electronic simulated code wheels and the message tape were available as inputs. The message tape appeared as five separate binary channels, as did each of the two main sets of code wheels. The Colossi also provided inputs representing the differences between two consecutive character positions and, in later models, five consecutive positions of one of the electronic code wheels. The results of these logical combinations could be fed into different electronic counters, the contents of which would be printed or not printed after the message was fully read according to thresholds set by

[31] Compendium, p. 136.

the user. After each reading of the message tape the initial code wheel settings would be stepped to a different combination according to switches configured by the user.

Operators plugged wires (or, in later models, set switches) to combine these inputs. This let them specify a truth table, making particular logical connections between the input pulses and the output fed to each counter. Colossus included circuits to implement a variety of Boolean logic operations, including XOR and NOT. By running pulses through several of these circuits other logical conditions could be specified. For example, input_1 OR input_2 could be implemented as NOT(NOT input_1 AND NOT input_2).[32]

Flexible as the logic circuits were, they performed Boolean logical comparisons rather than specifying algorithms or performing numerical computations. [33] Carrying out a computation involves sequencing operations over time, performing step after step after step. In contrast, the logical circuits of Colossus were wired to evaluate a complex logical condition in a single step. A particular combination of inputs was applied, and this either triggered an output pulse (representing a 1 in the corresponding truth table) or it didn't (a 0 in the corresponding truth table). The output was relayed to a counter, which either incremented or didn't increment. The logic circuits then reset, and processed the next combination of inputs. They performed one complex step repeatedly, but did not sequence operations or maintain any state information from one input character to the next.[34] Each time a character was read from paper tape a particular combination of inputs was fired and an output pulse was, or was not, generated to increment one of the counters.

Computer scientists use automata theory to distinguish between the fundamental capabilities of different kinds of automatic devices. The most advanced, including programmable computers from ENIAC onward, are equated with Turing Machines. Push down automata are less powerful than Turing machines, and finite state automata are less powerful than push down automata. As their name suggests, even finite state automata preserve state information from one operation to the next. In contrast, the "combining unit" of Colossus did not preserve state information and thus was not an automaton of any kind. The technical term for this kind of capability is "combinational logic" or "time-independent logic." That's a revealing term here as the combining unit performed only one evaluative step: inputs were applied and an electrical circuit was either completed, to output a pulse, or not completed, in which case no pulse was immediately output.

The network of logic gates configured on Colossus's combining unit by its user had the effect of implementing a logical truth table, specifying for each combination of true/false inputs whether or not a true output should be produced. Its function was in many ways analogous to that of a punched card tabulating machine, in that it considered a stream of inputs and incremented the appropriate counter

[32] GRT reprint p. 323.

[33] On a technical level the distinction might be challenged, as binary arithmetic is built from trivial logical operations. For example, adding two bits is a logical XOR combined with a carry. Which raises a question -–– could Colossus have been wired to do the carry part – output of one channel as input for another? That would only work if everything could be evaluated simultaneously.

The need to consider multiple characters of the input message was initially dealt with by building an electronic buffer so that two positions could be read simultaneously as input. In later versions of Colossus, a switch selected between inputs of actual message data or differences between consecutive character positions. Later versions of Colossus also buffered settings for one of the code wheels, so that up to five positions could be evaluated simultaneously. [34]

whenever the input data satisfied one of the conditions that had been set. Colossus allowed for rather more complex logical conditions than traditional tabulating machines, but it similarly transformed and counted streams of input data.

It has sometimes been claimed that Colossus possessed the capability for conditional branching, as a result of which computational mathematician S. Barry Cooper speculated that Colossus "may well be Turing complete" (i.e. could handle any problem solvable by any computer if given sufficient time and storage space).[35] Claims that Colossus implemented conditional branching seem more concerned with checking off a box in a comparison table than illuminating its actual architecture. Colossus certainly included circuits that would trigger an action only when particular data was read – for example resetting the machine when the end of message code was reached. But no configurable control connections could be made between different parts of the machine. Data pulses output from the combining unit were routed to counters, meaning that the only conditional action that could be specified using the logic circuits of the combining unit was to either increment or not increment each counter given particular inputs. This could not trigger any kind of change in the sequence of operations carried out. Once the end of the message was reached, other circuits would compare the total stored in the counter against the pre-selected threshold value to determine whether the totals and initial wheel positions should be printed. But Colossus could not carry out any other conditional action based on these totals, for example, by connecting a counter back to its uniservos so that the wheel start positions would change based on the value obtained in the previous cycle. No data of any kind, other than the uniservo settings, was retained in memory from one cycle of the message tape to the next.

We do not believe that the ability to reconfigure combinational logic to implement an arbitrary truth table should be considered evidence of "programmability" because it does not truly change the program, i.e. the sequence of operations performed by the machine. It merely provides a parameter used in one of those steps. Likewise the assertion made by Wells that different "algorithms" could be set up is misleading, since an algorithm expresses a computational procedure as a step-by-step series of operations carried out over time. Colossus was not programmable, according to our definition, because the basic sequence of operations performed, the program, could not be changed by the users.

Was Colossus Binary?

Colossus is often called a binary computer, sometimes with the implication that this made it close to modern computers, and hence in at least some respects more advanced, than computers such as ENIAC that used decimal number representations. Paul Gannon believed that because the "modern computer can be defined as an electronic, binary/logic-processing, conditional-programming, stored-program-control, general-purpose machine" and Colossus, unlike ENIAC, was also a "binary/logic-processing, programmable" machine "it can be seen that Colossus was closer to the modern concept of the computer than Eniac was in some significant ways."[36] Echoing this, in *The Innovators*, Walter Isaacson noted that "well before ENIAC… the British code breakers had built a fully electronic and digital (indeed

[35] Machines as Data, 2015, cite.
[36] Cite Gannon, circa p. 433.

binary computer." ENIAC, in contrast, "was not like a modern computer" to the extent that it used a decimal system.[37]

The connection of binary arithmetic with progress is rather dubious, as decimal arithmetic has some advantages for floating point and remained a feature of many later computer architectures.[38] The deeper question, though, is whether it makes sense to call Colossus binary at all. In popular use, binary is associated with the idea of representing information as a series of electrical pulses, conceptualized as 0s and 1s. Colossus certainly did that – its users spoke of "impulses" which corresponded to the "dots" and "crosses" of the teleprinter alphabet. However all digital electronic machines signal information as a series of pulses, this being the defining characteristic of "the digital." ENIAC, for example, transmitted a decimal digit as a series of pulses. The number four was transmitted as four pulses.

Computers stored and manipulated numbers, and what "decimal" and "binary" actually refer to is the base used to represent these numbers: 10 for decimal and 2 for binary. The difference between binary and decimal computers came in three main areas: the format used to store numbers in memory, the format used to transmit them, and the design of the arithmetic circuits used to manipulate them. ENIAC stored each decimal digit in a ring counter that could take one of ten possible positions. Each of its accumulators held ten of these "decades", and could therefore store a ten digit decimal number using what was, from the viewpoint of the electronics required to build it, 100 bits of storage. This was inefficient because only one of the ten bits for each digit would fire at a time. A binary computer could store the same number more efficiently, using 34 bits of storage. Almost all later computers therefore used binary representations to store numbers in memory, though many stored each decimal digit separately (using four bits per digit) and continued to use decimal arithmetic circuits to manipulate them. Computers stored and manipulated numbers of a fixed "word" length – 10 decimal digits for ENIAC, somewhere between 18 and 36 bits in many early digital machines.

Most of the information manipulated by Colossus did not represent numbers and Colossus performed logical transformations rather than arithmetic on its inputs. The message tape inputs represented characters, but for most jobs Colossus took only one or two of the five channels and did not directly consider the characters being encoded. The logical combination circuits did not store numbers at all. Colossus also had nothing comparable to a word length, as numbers were not stored, retrieved, or manipulated arithmetically. The inputs it processed were continuous.

It was able to count, i.e. increment a stored number. In fact three separate parts of the machine counted, each in a different way. The electronic rings that emulated the physical encoding wheels of the Lorenz machines were essentially a set of electronic counters of different lengths that advanced one position at a time. A set of electronic counters incremented each time an output pulse was received from the logic circuits in the combining unit, counting the number of matches achieved with the current combination of settings. The uniservos that reset the start positions of the electronic counters after each complete revolution of the message tape were themselves decimal counters advancing with each cycle

[37] Isaacson nevertheless considered ENIAC the most important of the early computers. Walter Isaacson, *The Innovators: How a Group of Hackers, Geniuses, and Geeks Created the Digital Revolution* (New York: Simon and Schuster, 2014), 75 & 79.

[38] Cite Kahan on floating point.

of the message tape. These used decimal and biquinary representations.[39] So in as much as Colossus continuously repeated non-numerical logical operations on bitstreams of arbitrary length it does not make sense to say it manipulated numbers in decimal or binary form. In as much as parts of Colossus were acting on numbers, those numbers were stored in biquinary form and manipulated in decimal form.

Conclusions – Why Should We Care About Colossus?

Colossus currently has a strange position in the historical literature on computing, lauded by its supporters but largely ignored in overview histories. It is not clear to us that "computer" is the most useful way of conceptualizing Colossus. We think that Flowers' own, more nuanced, characterization of Colossus as an "electronic processor" rather than a computer is essentially correct. Colossus was closer to a digital signal processor than a computer. Its core electronic units performed real-time logical transformations on two input bitstreams to yield an output bitstream (for the counters).

As Flowers proudly noted, Colossus certainly had many elements in common with early computers: counters, electronic logic, a printer and an input tape. But its architecture and purpose were fundamentally different. Colossus did not carry out computations, or have any capabilities for numerical operations. This sets it aside from every other machine celebrated in the history of computing. Colossus was a digital electronic device able to generate bitstreams electronically, to combine these bitstreams with others read from paper tape, apply logical transformations, and count the results. For decades, from the late-1940s to the 1990s, specialists and the public shared a fairly clear sense of what "the computer" was. It consisted of a box holding a processor and memory, to which were connected storage devices such as disk and tape and output devices such as printers. Over time the boxes shrank, they got cheaper, and they spread from data processing centers into schools, offices, and homes.

Computers were far more visible than other forms of digital electronics during the 1970s and 1980s, and invention of the computer was a much more visible honor than invention of the digital signal processor. Making a case for the historical importance of Colossus meant shoehorning it into this tradition, which is exactly what Brian Randell did when he won it a prominent place at the seminal 1976 Los Alamos museum where the early agenda for the history of computing field was set. In his earliest public statements on Colossus, Flowers himself saw his work as something quite different from computing and instead situated Colossus within the context of early work on digital communication. We think he was right, and that the modern proliferation of digital devices and the vanishing of "the computer" as a distinct and coherent thing makes us better able to appreciate Colossus on its own terms. In an age of digital audio and video, digital telephony, and digital data communications we can better appreciate the importance of this development and see it, as Flowers did himself, more as a contribution to the development of digital communication than to computing. It challenges us to broaden our focus, beyond the traditional reading of the history of computing in the 1940s as a series of incremental steps toward the "modern computer." If there has been such a thing as a digital revolution then it involved more than just computers, making Colossus an exemplary artifact of the early digital because of, not despite of, its stubborn lack of architectural resemblance to a computer.

[39] Biquinary representation, as used in the Colossus counters, took advantage of vacuum tube circuits that had not just two stable positions but five. Using a two position counter and a five position counter for each decimal digit allowed for ten possible combinations,

Having more precisely defined and historicized the concepts of program and programmability, we also challenge the prevalent concept of Colossus as "programmable." As we saw, Colossus's metaphorical trophy comes as the first "programmable electronic" computer, setting it aside from the slightly earlier "Atanasoff Berry Computer", which is said to have been electronic but not programmable. Colossus certainly followed a program in that it contained dedicated control facilities to sequence different operations over time. But it was not, contrary to previous claims, programmable in that this sequence could not be changed in any fundamental way. Configuring Colossus involved setting parameters, but not defining new sequences of operations. In this it resembled the Atanasoff Berry Computer, which was designed to accept parameters but followed the same basic series of steps to solve systems of linear equations.

Early computing innovation is often discussed from the viewpoint of logic, stressing the mathematical and theoretical insights behind the development of computer architecture.[40] The progression from one early computer to the next has been depicted as a series of abstract architectural innovations – first conditional branch, first stored program, and so on. In turn these innovations have sometimes been represented as mere practical instantiations of the work of Alan Turing on the mathematical logic of computation.[41]

Colossus matters, but claims for its historical importance can and should be about more than the exact string of adjectives to insert between "first" and "computer." Broadening our frame of reference to include other, recently declassified, wartime cryptography projects as well as the well-known computing projects of that era suggests that Colossus was unique among the many wartime projects as a digital electronic device employing unproven technologies that was built quickly enough and worked reliably enough to make a significant contribution to the war effort.

Colossus worked, and unlike many even of the successful projects was designed and built very rapidly. Many formerly secret electronic and microfilm-based machines discussed by Colin Burke in his recently declassified book on the era, like the "Freak", never worked reliably enough to be put to work on the tasks for which they were constructed. Some, like the Superscritcher (and, among computers, ENIAC) worked but were ready only after the war had been run. Most machines delivered before the end of the war used proven electro-mechanical technologies. This made a tradeoff between risk and performance. Yet even this did not guarantee they would be useful – Madame X, built by Bell Labs to tackle the Enigma code, was physically a much larger machine than Colossus and was wonderfully reliable, but design limitations caused by a lack of cryptologic understanding meant that it was of little use to the war effort.

Lengthy delays plagued most computing projects during and after the war. Archival research has convinced us that Colossus cannot be entirely separated from work on the "Robinson" series of machines, but even so less than a year went by between the request from Newman for Dollis Hill to work on what became Heath Robinson and Colossus and the delivery of the first Colossus to Bletchley Park.

---

[40] The classic contribution of this kind is Martin Davis, *Engines of Logic: Mathematicians and the Origin of the Computer* (New York, NY: Norton, 2001), .

[41] This assumption is critiqued in Thomas Haigh, "Actually, Turing Did Not Invent the Computer," *Communications of the ACM* 57, no. 1 (2014).

In its proper context, Colossus reminds us of something else: the enormous importance of professional engineering work in distinguishing the relatively small number of successful projects from the greater number of failures. This point comes across clearly enough in the established history of computing literature. The most successful of the wartime and pre-War machines were built by well-established engineering groups. The Harvard Mark I was built by IBM and drew on its long experience manufacturing punched card machines. Bell Labs, which built a series of tape-controlled relay calculators, benefitted from the Bell System's world leading work in building reliable, automatic telephone switching systems. ENIAC was built at the Moore School of Electrical Engineering and benefitted from the experience of its faculty and students as well as from connections to the local electronic industry (and to Bell Labs, which provided its relays and IBM, which provided its punched card peripherals).

In contrast, some other well-known early computing projects were constructed without access to professional engineering networks and suffered. Konrad Zuse built his first mechanical computer, a one-ton monster later dubbed the Z1, in his parents' apartment immediately before the war. It had many novel architectural features, but when he turned it on it seized up almost immediately. A replica he oversaw decades latter suffered the same fate. Andrew Booth built several early computers on shoestring budgets, but it is not clear whether his first, the Automatic Relay Computer ever worked at all in its original form.[42] The Atanasoff Berry Computer was built by an ingenious team and incorporated many novel features, but its homebrewed input/output system (using sparks to burn paper) never worked reliably enough to tackle the problems it had been built to solve. Unlike the successful projects of the era its creators could not draw on experienced corporate engineering teams familiar with paper tape or punched card storage.

Colossus also stood out for its reliability. Most electronic computers of the 1940s and early 1950s seem to have spent a year or so between being finished and being reliable enough to carry out useful work. ENIAC's creators were able to convince the patent office and several courts that ENIAC was not sufficiently debugged to do any useful work until July 1946, more than six months after its first test use. After being moved to Aberdeen at the start of 1947 it relapsed into unreliability, and for more than a year struggled to carry out any useful work. Colossus, in contrast, was handling production work by March 1944.[43]

Like the Bell Labs team that worked on Madame X, Flowers and his colleagues were part of a telecommunications engineering institution. They could draw on experience, internal engineering talent, and existing relationships with component suppliers. Yet, unlike the more conservative designs of Bell Labs, Colossus made extensive use of electronics for counters and logic. Flowers has insisted that these technologies were not unproven to him, even if the rest of the world was skeptical, because of the pre-War work he had been doing on electronic telephone switching. This commitment to high performance engineering extended to the tape drive, which determined the performance of Colossus as a whole. Dollis Hill drew on the expertise of F.O. Morell and other engineers in the Telegraph Group for help with the engineering needed to move paper tape at unprecedented speed, and of course on the experience

[42] I recently saw a reference that Booth sought IO for a later machine which led to some kind of commercial development, so will be careful here.

[43] 1944 Mar 29, HW 62/6, De Grey to Radley notes Colossus 1 is "now shortly coming into full operation."

of Flowers in building electronic sensors of the tape drive (used with the Heath Robinson prototype) and additional electronic units for Colossus.[44]

Flowers benefitted from his existing partnership with the codebreakers at Bletchley Park, which gave him the feedback needed to optimize the usefulness of his machines. Experiences with Heath Robinson shaped the capabilities of Colossus, and experience with the first Colossus led to substantial modifications for its successors. Not only did Flowers lead a project to quickly build a reliable machine from exotic parts, but he also built exactly the machine that was needed by Britain's codebreakers to deliver intelligence to Allied leaders.

This underlines the need for a historiography of the early digital that is concerned with use rather than just invention, and for a history of computing fully integrated with broader historical analysis, such as social, business, labor, and military history. From the viewpoint of programmability, though not of utility or flexibility, Colossus seems to us fundamentally similar to another well-known special-purpose digital electronic computer: the Atanasoff Berry Computer. The ABC was likewise driven by the rotation of a physical medium, in this case a rotating capacitor drum memory rather than the paper tape used with Colossus. It carried out a sequence of mathematical operations, performing some steps on a conditional basis depending on the results obtained. Like Colossus the basic sequence of operations performed was fixed, but its behavior would change based on the parameters supplied when the machine was configured by setting switches and rewiring plug boards.[45] ABC and Colossus have traditionally both been granted "firsts" which were distinguished as follows: the ABC was a special purpose non-programmable digital electronic computer whereas Colossus, built a few years later, was also an electronic digital computer but was in addition programmable.

For the traditional perspective on the historiography of each computing, the only thing separating Colossus from the ABC is the questionable insertion of the word "programmable" between "first" and "electronic computer." We should perhaps be more impressed by the fact that one of these machines was never able to carry out the work it had been built to do and was therefore abandoned in a basement, while the other was spectacularly successful in use. The ABC's logic and memory units worked well, but to solve the equations it was designed to work with it also needed a memory to store intermediate results. The paper based storage solution its creators came up with, burning paper sheets to store bits, never worked reliably enough for the machine to tackle the problem it was designed to handle: solving systems of up to twenty-nine simultaneous linear equations. It did nevertheless do some useful work on much simpler statistical problems, but these were relatively trivial and would never have justified the creation of such an elaborate machine.

Colossus, in contrast, proved highly effective when applied as intended and in fact proved able to carry out a broader range of cryptographic tasks than originally intended. In doing so it made an appreciable contribution to the course of the war. The precise impact of codebreaking on the outcome of the war is hard to quantify. It's often claimed that breaking Enigma, or sometimes even the personal contributions

[44] Copeland, p.295, attributes to Fensom the idea that Flowers got involved only after Morell's telegraph group ran into difficulties. We haven't verified that with primary sources, but 1943 Mar 12, HW 14/70 does note that Flowers and Morell were visiting Newmann together do discuss plans for what became Colossus as well as what became Heath Robinson.

[45] Cites on ABC.

of Alan Turing, shortened the war by two years.[46] We find that unconvincing, as did Max Hastings and John Keegan in overview histories of the contributions of intelligence work.[47] Yet Colossus facilitated the reading by Allied commanders of some of Nazi Germany's highest level military communications, including messages written by Hitler himself. It has been credited with a crucial role revealing German troop positions and plans in Normandy before and after the D-day landings, so that Allied troops could avoid German forces when landing and then head off attempted counterattacks in the months that followed. If Colossus shortened the war by even a month, that would still make it one of the most consequential machines in history. That, surely, is a more important basis on which to characterize its importance to history.

Burks, Alice. *Who Invented the Computer: The Legal Battle that Changed Computing*. New York, NY: Prometheus Books, 2003.

Campbell-Kelly, Martin, and William Aspray. *Computer: A History of the Information Machine*. New York, NY: Basic Books, 1996.

Ceruzzi, Paul E. *A History of Modern Computing*. Cambridge, Mass.: MIT Press, 1998.

Copeland, Jack. "Colossus and the Rise of the Modern Computer." In *Colossus: The Secrets of Bletchley Park's Codebreaking Computers*, edited by Jack Copeland, 101-15. New York: Oxford University Press, 2006.

Davis, Martin. *Engines of Logic: Mathematicians and the Origin of the Computer*. New York, NY: Norton, 2001.

Eckert, J Persper, John W. Mauchly, Herman Heine Goldstine, and John G Brainerd. *Description of the ENIAC and Comments on Electronic Digital Machines. AMP Report 171.2R. Distributed by the Applied Mathematics Panel, National Defense Research Committee, November 30*. Philadelphia, PA: Moore School of Electrical Engineering, 1945.

Flowers, Thomas H. "Colossus and the Rise of the Modern Computer." In *Colossus: The Secrets of Bletchley Park's Codebreaking Computers*, edited by Jack Copeland, 91-100. New York: Oxford University Press, 2006.

———. "The Design of Colossus." *Annals of the History of Computing* 5, no. 3 (Jul-Sep 1983): 239-52.

Gannon, Paul. *Colossus: Bletchley Park's Greatest Secret*. London, UK: Atlantic Books, 2006.

Grier, David Alan. "The ENIAC, the Verb "to program" and the Emergence of Digital Computers." *IEEE Annals of the History of Computing* 18, no. 1 (January 1996): 51-55.

[46] The claim is often made that the Second World War was shortened by two years either by (a) Alan Turing's personal contributions to codebreaking, (b) breaking Enigma, or (c) Colossus. The source of the idea seems to be the introduction to E F Hinsley and Alan Stripp, eds., *Code Breakers: The inside Story of Bletchley Park* (New York: Oxford University Press, 1993) where it is made for Ultra intelligence, i.e. the Bletchley Park operation as a whole. Specifically, the authors suggest that without the benefit of Ultra, the battle of the Atlantic and the campaign in North Africa would both have taken significantly longer to win, with the result that the D-Day landings could not have been attempted in 1944. They also believe that the landings would then have failed or been impossible in 1945 without Ultra intelligence and during attack by V weapons and new weapons that in our timeline Germany was not able to bring into operation. Thus D-Day would not take place until 1946, lengthening the war by two years. This rather Britain-centric view seems to us to neglect the overall trend of the conflict, particularly the rapid progress of Soviet troops westward during the first half of 1944, suggesting that the Red Army would have reached Berlin well before mid-1947 with or without the D-Day landings. We also note that the atomic weapons used in Japan in 1945 could have been deployed against Germany if the war had lasted much longer than expected.

[47] Max Hastings, *The Secret War* (New York, NY: Harper Collins, 2016),

———. "Programming and Planning." *IEEE Annals of the History of Computing* 33, no. 1 (Jan-Mar 2011): 86-88.

Haigh, Thomas. "Actually, Turing Did Not Invent the Computer." *Communications of the ACM* 57, no. 1 (January 2014): 36-41.

Haigh, Thomas, Mark Priestley, and Crispin Rope. *ENIAC In Action: Making and Remaking the Modern Computer*. Cambridge, MA: MIT Press, 2016.

———. "Reconsidering the Stored Program Concept." *IEEE Annals of the History of Computing* 36, no. 1 (Jan-Mar 2014): 4-17.

Hastings, Max. *The Secret War*. New York, NY: Harper Collins, 2016.

Hinsley, E F, and Alan Stripp, eds. *Code Breakers: The Inside Story of Bletchley Park*. New York: Oxford University Press, 1993.

Isaacson, Walter. *The Innovators: How a Group of Hackers, Geniuses, and Geeks Created the Digital Revolution*. New York: Simon and Schuster, 2014.

Randell, Brian. "The Colossus." edited by N Metropolis, J Howlett and Gian-Carlo Rota, 47-92. New York: Academic Press, 1980.

Smith, Michael. *The Secrets of Station X: How the Bletchley Park Codebreakers Helped Win the War*. Biteback Publishing, 2011.