All About Squeeze-Keying (Part 2)

by Karl Fischer, DJ5IL

The 2nd and final part of the story takes us from straight keys and bugs, through the ultimatic mode to modern iambic keyers. The theory and development of these innovations are discussed in detail...

Single-dot mode

IN JUNE 1965 **Edward B. Brown, WØEPV**, submitted a patent application describing a *"Telegraphic Keyer"* which was approved by the United States Patent Office in February 1968. His twin-lever keyer is obviously a variant of the ultimatic, but contrary to that it has a lever-actuated time-base so that characters start immediately when a lever is hit. According to the patent paper it requires *"fewer and less precise manipulations [...] as compared with those of the art to date"* due to its unique feature of a *"single-dot- injection storage circuit"*, and that's why I will call its keying logic the "single-dot mode".

The keyer works like the twin-lever Ultimatic with two variations: 1) it does not have a dash-memory and therefore if a dash is to be inserted or appended the dash-contact must be held closed until the dash starts and 2) when the dot-lever seizes control while the dash-contact is held closed, only one single dot is inserted after completion of the current dash and then the keyer reverts to dashes until the dash-lever is released or the dot-lever makes contact again. Like the Ultimatic it has a dot memory which is cleared already at the *onset* of the dot-element and so a dot can be stored at any time, even during a dot-element. But contrary to the Ultimatic the dot-lever loses control after one single dot.

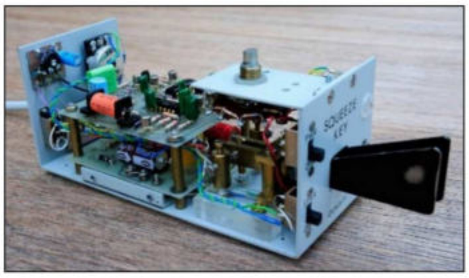
For example, to key a "C" you could press the dash- lever for the first dash and hold it. Then you could flick in the two dots like in ultimatic mode, which requires a total of three strokes. However, this would violate the squeeze-keying rule which results in the following correct technique for this mode: keep also the dot-lever pressed after the first dot, then release only the dash-lever before the last dot starts and finally release the dot-lever, which requires a total of only two strokes.

The single-dot mode is characterized by the following properties: The stroke rate is 1.78 (64 / 36) and the hold time is 9.7 (618 / 64) exactly like in ultimatic mode, the "C" needs one stroke less and the "X" one stroke more. The persistence is slightly reduced to 78% because 8 characters starting with a dash (B C D X Z 6 7 8) do not allow to keep the initial dash-lever pressed, but on the other hand it gives more time to release the dot-contact after insertion of a single dot between two dashes (C K Q Y). Neglecting those characters which require just a single stroke, all other letters of the alphabet except for the "X" and all other digits can be keyed with just one closure of each lever contact which equals two strokes. So the patent's claim that this keyer requires fewer and less precise manipulations is certainly unjustified.

The single-dot mode works like a cross between the ultimatic mode and the iambic mode, which had not been invented at that time and which is explained later. Indeed, an operator who is used to plain iambic mode should have no problems at all to operate in single-dot mode, if he only minds to release the dash- lever before the trailing dot(s) of the letters "C" "F" "L" "R" start(s). The same applies to the ultimatic mode, but the letters "C" and "K" are a bit difficult to key for an iambic operator and provoke errors.

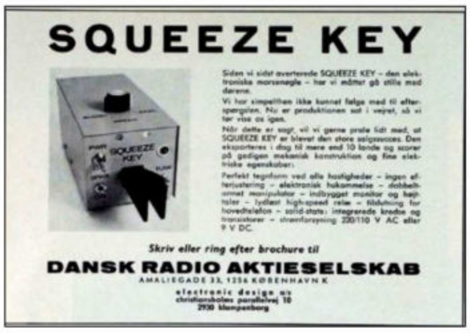
Jimmy Moss, W5GRJ, presented *"The WØEPV Squeeze Keyer"* in his article of July 1967 [12] as *"Ed Brown's remarkable gadget that has kicked up a right smart bit of talk on the c.w. bands"*. Among the readers was a very interested old acquaintance: Bo, OZ7BO, the CW nut from Denmark, who already had proved his nose for good keyer designs as well as his skill and expertise to make them even better. But contrary to some 20 years ago, when he had published his el- bug design in the "OZ" magazine, this time he was obviously determined to exploit his abilities.

Bo's son Ole owned a company named "Quali-Fi", a consultant in audio engineering and distributor of professional studio & broadcast sound systems and high-end hi-fi equipment in Scandinavia, which still exists under new management. So OZ7BO started the design and construction of a new keyer in the lab of Quali-Fi, transforming the original circuit of WØEBV with 4 valves and one transistor into a state-of-the-art circuit with 9 transistors and two RTL (Resistor-Transistor-Logic) ICs on a small printed circuit board and building it together with a power supply board and a dual-lever key into a small grey cabinet. A product review appeared in the March 1968 issue of "OZ" and then Quali-Fi helped him starting up with a first production run of 25 units of that "SQUEEZE KEY" type MSK-4, which were already sold before made.



This picture shows my own MSK-4 with the cover removed, I bought it in 2018 from the estate of SM3DNI for about 25 Euros and totally disassembled and refurbished it. Except for some minor modifications it is in original condition and a pleasure to operate.

Production went on, but as Bo could never stop improving the keyer it was decided that he should form his own company "electronic design a/s". He replaced the RTL with TTL ICs and revised the circuit accordingly, integrated power-supply and logic into one single circuit board and replaced the stepwise with a continuous speed control.
Advertisements appeared and a brochure was available on request from "Dansk Radio



SQUEEZE KEY

Siden vi sidst averterede SQUEEZE KEY – den elektroniske morsenøgle – har vi måttet gå stille med dørene.

Vi har simpelthen ikke kunnet følge med til efterspørgslen. Nu er produktionen sat i vejret, så vi tør vise os igen.

Når dette er sagt, vil vi gerne prale lidt med, at SQUEEZE KEY er blevet den store salgssucces. Den eksporteres i dag til mere end 10 lande og scorer på gedigen mekanisk konstruktion og fine elektriske egenskaber:

Perfekt tegnform ved alle hastigheder – ingen efterjustering – elektronisk hukommelse – dobbeltarmet manipulator – indbygget monitor og højttaler – lydløst high-speed relæ – tilslutning for hovedtelefon – solid-state: integrerede kredse og transistorer – strømforsyning 220/110 V AC eller 9 V DC.

Skriv eller ring efter brochure til

DANSK RADIO AKTIESELSKAB
AMALIEGADE 33, 1256 KØBENHAVN K

electronic design a/s
christiansholms parallelvej 10
2930 klampenborg

Aktieselskab", the Danish Radio Inc. which was formed in 1920 by the leading Danish shipping companies. The table in that brochure showing lever timing in the formation of various characters is an almost identical replication of Fig. 4 in W5GRJ's

QST article, and the *"single-dot- injection storage circuit"* featured in WØEPV's patent paper is now called *"single dot memory and injection system"*. The brochure explains: *"You will find that the dash lever overrides the dot lever, causing the keyer to shift to dashes. This means that in sending any character starting with a dot, the dot lever need not be released until the end of the character. An important feature of the SQUEEZE KEY is its single dot memory and injection system which makes the keyer capable of forming any letter except X with only one closure of a double lever key."*

According to his son Ole's guess about 500 to 1000 units of the new type MSK-5 keyer were built until Bo died in 1972 and his business was sold to a Swedish company in Malmø. During the 1970s the famous US company "Henry Radio" also marketed the "Tempo" line of equipment, which was supplied by various manufacturers. They also advertised the "TEMPO DKT ELECTRONIC KEYER" in amateur radio magazines as *"the latest in electronic keyer design"*. In fact, it was an MSK-5 with a black instead of a grey cabinet and modified marking, and it is quite likely that "DKT" stands for the Danish company "Dansk Kabel Teknik" which still exists today with a branch in Malmø. This keyer was offered only for a very short time, so obviously DKT had sold the remaining batch of MSK-5 units with a modified outfit to Henry Radio but did not continue its production.

While neither the Ultimatic keyer of W6SRY nor the WØEPV keyer gained currency and were largely eclipsed by the iambic mode, which came up in the late 1960s, OZ7BO's "SQUEEZE KEY" became very popular and widespread only in Scandinavia among radio amateurs as well as professionals. It was also used at the coastal radio station "Blaavand Radio" with the callsign OXB, located on the west coast of Jutland and noted for its characteristic low-pitched modulated CW (mode A2) signal. This photo was taken in 1980, it shows radio telegraphy operator Connie Nielsen at her OXB operating position. Note the MSK-5 keyer together with a straight key on the desk.

The bottom line is that OZ7BO did not really create the single-dot keying mode of his "SQUEEZE KEY". But he sparked its vast popularity in Scandinavia by converting the original WØEPV keyer - which was a valve circuit and hence already outdated in 1968 - into a compact and modern solid-state device, producing it in numbers and selling it for an affordable price.

Iambic mode

The name of this twin-lever keying mode is derived from the *iambus,* a metrical foot in poetry with alternating short and long syllables like "dah-di-dah-di-dah". In January 1967 **Harry Gensler Jr., K8OCO**, described his *"Iambimatic"* keying concept [13]. In the introductory words he recaps that *"Letters which contain dots alternated with dashes (requiring a back- and-forth* *motion) have encouraged the use of memories in more complex keyers, such as the Ultimatic and Penultimatic"*, and he expressly states that contrary to these keyers his concept *"uses no memories"*.

We will come back to this statement later when we challenge the usefulness of dot/dash-memories, but for now it is important to note that they are *not* used in the original keying-mode invented by K8OCO. I will sometimes call it the *plain* iambic mode in order to distinguish it from its two variants iambic type A and type B. The keying concept of K8OCO may be applied to practically any keyer and he presents three examples: an Iambimatic modification of the Hallicrafters HA-1 ("TO") single-lever keyer, a complete Iambimatic keyer, and a universal modification design which converts any simple single-lever keyer into an Iambimatic keyer.

While only one lever-contact is held closed it generates a string of dots or dashes, exactly like any single-lever keyer or twin-lever keyer operating in ultimatic or single-dot mode does. However, squeezing both levers generates a string of dot- and dash- elements in *alternating* sequence with the initial element corresponding to the lever which was hit first. Plain iambic keying can be generated by executing this simple set of instructions:

poll both levers alternately, if the lever is pressed generate the corresponding element and continue polling.

So, if the squeezed levers are released while an element is in progress, a plain iambic keyer without dot/ dash-memory simply completes that element. For example, to key a "C" the dash-lever is pressed first, immediately followed by the dot-lever, and both squeezed levers are released any time during the last dot-element.

The iambic mode is characterized by the following properties: The stroke rate is 1.81 (65 / 36) and the hold time is 9.0 (588 / 65), the persistence is substantially reduced to 64% because the 13 characters containing an embedded or trailing series of two or more identical elements (B D J P W X Z 1 2 3 6 7 8) do not allow to keep the initial lever pressed. Neglecting those characters which require just a single stroke, all other letters of the alphabet except for the "P" and "X" and all digits can be keyed with just one closure of each lever-contact which equals two strokes.

In 1967 Hermann Samson, DJ2BW, started to manufacture his ETM series of electronic keyers, developed by radio telegraphy instructor Klaus Duhme from the maritime school in Elsfleth / Germany. Their first twin-lever iambic squeeze keyer was the model ETM-3 which appeared around 1970. ETM keyers became very popular among radio amateurs in Germany and neighbouring countries and were also used by the maritime as well as other "special" radio services.

Operating according to the original K8OCO keying concept they had no dot/dash-memory until 1983 when the model ETM-5C appeared, and so I learned squeeze-keying in original plain iambic mode in the early 1970s as a teen-aged novice radio amateur. It forced me to ingrain proper element sequence timing within characters and also helped me to develop the feeling for proper inter-character and inter-word spacing. However, asking fellow radio telegraphy operators it turns out that today this seems to be a most unusual keying mode (see the survey presented later).

The Curtis-keyer

The first iambic keyer which appeared on the market in 1969, the *"Electronic Fist"* EK-38 by John Curtis, K6KU, already extended that plain iambic logic by a *dot-memory*. In the 1960's John worked for Signetics and he decided that a complete keyer could be implemented on a chip. In 1973 Curtis brought out the 8043 CMOS chip, the first integrated-circuit iambic keyer with dot-memory.

Of course, no logic is able to foresee and hence it is impossible to compensate for our finger movement being too slow - but logic can remember and so it can compensate for being too quick. The behaviour of the Curtis-keyer can be emulated by the basic iambic set of instructions together with the following dot-memory rule:

if any time during generation of a dash-element the dot-lever was hit, generate one extra dot-element.

To accomplish this, the dot-contact is polled not just for its state and not only when no element is in progress as in plain iambic mode, but for a *change of state from open to closed* and constantly while a dash-element is generated. That change of state is remembered until the dash-element is completed, and then one extra dot-element is generated and the memory is cleared. If the dot-lever is hit but not released during a dash-element the memory is unnecessarily set, because a following dot-element would still be triggered by the pressed lever.

Note that the dot-memory is cleared only at the *end* of the dot-element and therefore a dot can be stored only during an opposite dash-element but not during a dot-element, while in ultimatic and single- dot mode the dot-memory is cleared already at the *onset* of the dot-element and therefore a dot can be stored even while a dot-element is generated.

The sole purpose of the dot-memory is to increase the dot-lever timing leeway by allowing its too early pressure and release (more on that later). However, when the keyer is manipulated with proper timing it should behave exactly like a plain iambic keyer with the same stroke rate of 1.81, hold time of 9.0 and persistence of 64%. To show you how Curtis' dot-memory works and that it perfectly meets this requirement, suppose you want to key an "N": You press the dash- lever first to start the dash-element. Then you can either release it and press the dot-lever (single-lever keying) or hold it and also press the dot-lever (squeeze keying). Then you can either release the lever(s) before the dash-element is completed and the keyer will append a dot-element to key the "N" by utilizing its dot-memory. Or you can hold the lever(s) until the dot-element is in progress and then release to key the "N" with proper timing as in plain iambic mode. So there are four possible keying methods to get an "N" out of the Curtis-keyer.

The Accu-keyer

This iambic keyer by James Garrett, WB4VVF, featuring dot- and dash-memory as well as automatic inter-character spacing, was published in 1973 [14] shortly after John Curtis' 8043 chip appeared. The dot/ dash-memories of the original circuit are TTL latches which are set always when

the related lever-contact is closed and cleared only if that lever-contact is open at the end of the related element, therefore a dot or dash can be stored only during an opposite element. Not the levers directly but only their memories are polled in order to decide which element is to be generated. Therefore the whole operation of the Accu-keyer is based on its memory logic, and that's why it must have both a dot- and a dash-memory. Pressing and holding only one lever produces a string of corresponding elements, when it is released, its memory is cleared immediately at the end of the current element and no additional element is generated. However, if both squeezed levers are released together the memory opposite to the current element is not cleared and remains set, and therefore one surplus alternate element is generated.

The behaviour of the Accu-keyer can be emulated by the basic iambic set of instructions together with the following dot/dash-memory rule:

if anytime during generation of an element the opposite lever was pressed, generate one extra alternate element.

So, neglecting the fact that the Accu-keyer has both a dot and a dash-memory, the only procedural difference is that while an element is generated it just re- members the *state pressed* while the Curtis-keyer remembers the *change of state* or *transient from unpressed to pressed* of the opposing lever, and if the memory is set both keyers generate one extra alternate element.

Though this subtle difference between both memory rules at first glance seems negligible, it has a profound and detrimental side-effect: while the Accu-keyer offers the same timing leeway as the Curtis- keyer when hitting a lever, contrary to that it does not allow to hold both levers squeezed as long as possible with proper timing in plain iambic mode. Because if you do so, it remembers a pressed lever and generates an unwanted extra alternate element.

This problematic memory logic affects all characters which require more than one stroke, or in other words, all characters which are squeezed because they contain a dot + dash or dash + dot sequence. With a plain iambic or Curtis-keyer *both squeezed levers must be released only before the last space of that sequence ends*, so you have plenty of time to release both levers and the higher the keying speed the more beneficial that leeway is. However, with an Accu-keyer *one of both squeezed levers must be re-leased already before the last dot or dash of that sequence starts*, so that you have much less time . The most problematic of these characters is the "A" which frequently becomes an "R" if the dot-lever is not re-leased already during the short initial dot-element. Less frequently, because there is more time to release the dash-lever during the long dash-element, an "N" becomes a "K" or a "K" or "D" becomes a "C".

The maximum time window allowed to hold a squeeze is identical for the plain iambic and the Curtis-keyer, but for the Accu-keyer it is reduced by the length of the last element of that sequence. For example, at a speed of 30 WPM a dot or space is 40 ms and a dash 120 ms long. Squeezing an "A" you have 240 ms (dot + space + dash + space) to release both levers with a plain iambic or Curtis-keyer but only 80 ms (dot + space) or 33% thereof to release the dot-lever with the Accu-keyer, otherwise you get an "R". And keying a "K" you have 400 ms to release both levers with a plain iambic or Curtis-keyer but only 240 ms or 60% thereof to release the dot-lever with the Accu- keyer, otherwise you get a "C".

Compared with the Curtis-keyer the Accu-keyer has the same stroke rate of 1.81, but because one of both squeezed levers must be released earlier and only 14 characters (E F H I K L M O Q R S T Y 5) allow to keep the initial lever pressed, the hold time and persistence are both substantially reduced from 9.0 to 7.8 and from 64% to 39 % respectively.

It follows that the Accu-keyer does not behave like a plain iambic keyer when it is manipulated with proper iambic timing. And its dot/dash-memory logic in fact does not increase timing tolerance, but instead even boosts the possibility for errors with increasing speed because it requires a much faster release of a squeeze. In view of this analysis, I think it is fair to call the Accu-keyer's dot/dash-memory logic a major design flaw. Nevertheless, WB4VVF has sold thousands of printed circuit boards and his keyer became so popular - especially in the USA - that its dubious behaviour was adopted for keyers made by major manufacturers of amateur radio equipment.

Iambic type A and B modes

In 1975 the 8044 chip was introduced by John Curtis, an improved 8043 with dot- and dash-memory. At that time most CW operators already used iambic keyers - but scarcely anybody in plain iambic mode without dot/dash-memory, because neither the Curtis-keyer nor the Accu-keyer allowed to disable that feature. So, over the years two schools of iambic keying developed, differing only in the dot/dash-memory logic which the operators initially learned but rarely scrutinized or even changed: *Curtis-keyer* and *Accu-keyer*. In light of that fact, Curtis named his own logic *type A* and that of the Accu-keyer *type B* and in 1986 he introduced the 8044ABM chip which operated in selectable type A or B iambic mode.

Those who learned iambic type A (Curtis-keyer) are usually unable to master iambic type B (Accu-keyer) and vice versa. And while both groups are usually unable to master the plain iambic mode, those who learned it should have absolutely no problem with type A. When both squeezed levers are released, a type A keyer simply completes the element in progress whereas a type B keyer generates an extra alternate element. That's how the difference between the two iambic mode types is usually explained and this simple explanation is true. But it is incomplete, because it only describes an effect of type B without explaining its single cause: the inferior dot/dash- memory logic of the Accu-keyer which was described in detail before. This annoying surplus element can be avoided by not squeezing or by using a single- lever paddle, but of course that is not the intended mode of operation for any twin-lever keyer. If the dot/ dash-memory in type A and type B could be disabled, which is normally not the case, both types would behave absolutely identical and boil down to plain iambic mode.

To test a keyer for dot/dash-memory and its iambic mode type, set the speed as low as possible and key an "N" as fast as possible - both levers must be released before the dash-element is completed! With dot-memory the dash is always followed by a dot and you get the "N", without dot-memory the dot is lost and you get a "T" instead. Now key an "A" as fast as possible - again, both levers must be released before the dot-element is completed! With dash-memory the dot is always followed by a dash and you get the "A", without dash-memory the dash is lost and you get an "E" instead. Without dot- and dash-memory the keyer works in plain iambic mode, with dot-memory squeeze a "K" and release both levers only during the second dash-element. If you get the "K" the keyer works in type A, if you get a "C" instead it works in type B mode.

Dot/dash-memory scrutinized

When a single dot is to be inserted or appended after a dash, the operator of a single-lever keyer must swing the lever to the dot-contact while the dash- element is still in progress and swing it back to the dash-contact or release it while the dot-element is still in progress. But the higher the speed the shorter the duration of a dot and the hold time and the more difficult is its timing. The operator is apprehensive because if he holds the lever too long at the dot- contact he gets two dots, and in order to prevent this he tends to be over-hasty so that the lever is already back at the dash-contact or released before the dash-element ends. The result is a lost dot and the cure is a *dot-memory*, which seemingly keeps a once closed dot-contact closed until the dot-element starts or ends. When more than one dot is to be inserted or appended the dot-memory is useless, because the dot-contact must be held closed anyhow by the operator until the last dot starts.

Keying all 26 letters of the alphabet and all 10 digits a single dot is inserted within four characters (C K Q Y) and appended at the end of seven characters (C F G N P R 9) after a dash, and so the described risk of a lost dot exists with 11 out of these 36 characters which is 31 % or almost a third. It follows that dot-memory substantially increases timing leeway with any single-lever keyer (or twin-lever keyer when operated not squeezing in single-lever style), while dash-memory is generally unnecessary because the three times longer duration of a dash makes its timing much easier and hence lost dashes are scarce. And that's why Dave Muir, W2VYO, implemented only a dot-memory in his Penultimate single-lever keyer.

We remember that the memory feature was originally developed already in 1953 by W6SRY and that it was an absolutely necessary key element of his Ultimatic designs, both the single-lever and the twin-lever version, because without a dot-memory *and* a dash-memory the continuously running time-base would have caused dropping of leading elements. With a modern keyer operating in ultimatic mode dropping of leading elements is not a problem any more, and hence a dot- and dash-memory is not absolutely necessary. However, this mode behaves in a way similar to a single-lever keyer because when- ever a lever is pressed the subsequent elements correspond to that lever until it is released or the other lever is pressed. So, there is the same risk as with a single-lever keyer to get two dots instead of one and therefore dot-memory substantially increases timing leeway also in ultimatic mode.

How about single-dot mode? A dot-memory is certainly *not* necessary when a single dot is inserted (C K Q Y), because this mode automatically reverts to dashes after a single dot (hence its name), which means that the dot-lever must not be released already during the dot but only during the following dash- element. So there is no risk at all to get two dots instead of one and no need for the operator to be hasty. With four (C F P R) of the remaining seven characters where a single dot is appended at the end, the dot-lever is not hit shortly before the final dot but much earlier and it must be released only during the final dot. So the hold time is much longer and the timing easier than with a single-lever keyer and hence there is not much risk to get two dots instead of one. It follows that in single-dot mode the dot-memory increases timing leeway only with three characters (G N 9) which is a mere 8% of all 36 characters.

And finally, how about the most popular iambic mode? Concerning the timing of a single dot which is inserted or appended after a dash it behaves identical to the single-dot mode with one variation: dot- memory increases timing leeway with four (G N P 9) instead of three characters, which still is a meagre 11% of all 36 characters.

The foregoing analysis assumes that with any twin-lever keyer the squeeze-keying rule is strictly applied for maximum efficiency. However, unfortunately most operators are not consequently squeezing but instead treating their twin-lever keyer similar to a single-lever keyer and that's why they make accordingly more use of the dot-memory. If you are using a plain iambic or iambic type A keyer, watch your fingers carefully while keying a "Q" or an "F": do you really release both squeezed levers only during the last dash of the "Q" or the last dot of the "F" ? If your answer is *no* you do not follow the squeeze-keying rule, instead you flick in the dot of the "Q" and the dash of the "F" by briefly hitting the lever which means needless rush and the risk of a lost dot or dash in plain iambic mode.

The bottom-line is that with a twin-lever keyer in single-dot or iambic mode dot-memory does really make sense only if it is operated like a single-lever keyer, which means not to utilize the benefits of squeeze-keying and to blow the chance to learn and ingrain proper timing. Therefore it is my advice to all those who want to start with iambic keying to decide for the plain iambic mode without dot- or dash- memory.

Comparing the modes

After this thorough analysis of the squeeze-keying modes we are finally able to compare their efficiency. The following table lists for all keying modes their **year** of appearance and stroke rate **S**, and for the squeeze-keying modes also their hold time **H** and *persistence* **P** provided that the operator is consistently squeezing:

keying mode	**year**	**S**	**H**	**P**
straight	1844	3.67	-	-
sideswiper	1904	3.67	-	-
bug	1905	2.78	-	-
el-bug	1940	2.03	-	-
ultimatic	1955	1.78	9.7	100%
single-dot	1967	1.78	9.7	78%
plain iambic	1967	1.81	9.0	64%
iambic type A	1969	1.81	9.0	64%
iambic type B	1973	1.81	7.8	39%

This table reveals a very interesting development: starting with the sideswiper of 1904, which had the same average stroke rate as the straight key with 3.67 strokes per character but allowed for much higher keying speed with less physical strain, the stroke rate decreased and hence the efficiency improved with every new mode. It culminated in the first squeeze-keying ultimatic mode of 1955 with a stroke rate of 1.78, less than half of the sideswiper's value. However, with the following iambic modes the stroke rate did not decrease further but even slightly increased to a value of 1.81, and the average hold time and/or the persistence decreased substantially so that the overall efficiency deteriorated. It follows that squeeze- keying is indeed most effective in the ultimatic mode, followed by the single-dot mode which is very easy to master for a plain iambic operator, while it is least effective in the latest iambic type B mode.

In 2015 I asked an international group of dedicated and proficient amateur radio telegraphy operators for their preferred electronic keying mode and received 68 answers, here is the result of my survey:

25 = 36.8% iambic type B
21 = 30.9% el-bug
14 = 20.6% iambic type A
7 = 10.3% plain iambic
1 = 1.5% ultimatic

Interestingly the most effective squeeze-keying ultimatic mode is by far the least common, while most operators prefer the least effective squeeze-keying mode iambic type B despite its problematic dot/dash-memory logic. And it turned out that the preference of most of them is indeed not the result of an experimenting process with a final decision for the most effective and personally most suitable mode, but it is simply the mode they initially learned and never changed ever since. Once a certain keying mode is ingrained, it seems really hard to give it up for a better one ...

Variants and phonies

As described before, the short squeeze release time window of iambic type B mode is most problematic for the dot-memory and makes it almost impossible to squeeze for example the character "A" at high speed. The creators of the *CMOS Super Keyer*, which appeared in 1981 [15] and became very popular, were well aware of that fact and therefore they implemented this timing variant: *the dot- and dash-memories generally work according to type B, but during the first dot-length (first third) of a dash the dot-memory is disabled so that it cannot be set by a pressed dot- lever.* This gives the operator more time to release a squeeze than in true type B, but the drawback is that the insertion of a dot by tapping the dot-lever may fail: key an "N" as described in the test before and you will often get a "T" instead, indicating a disabled dot-memory. From the firmware version 2.0 on, which appeared in 1991, the Super Keyer can be set to emulate other modes in addition to that default setting "V0" which is designated as Super Keyer (or Logikey) timing w/dot and dash memory. Unfortunately, in contrast to the Super Keyer with its unmistakable mode designations there are also quite a number of phonies, keyers which pretend to work in type A or type B mode but which actually do not.

One prominent example is the internal keyer of all *Elecraft* transceivers and what is called "mode A" and "mode B" are in fact both timing variants and mixtures of the true type A and type B modes. *In Elecraft's "mode B" the dot- and dash-memories generally work according to type B, but during the first dot-length (first third) of a dash the dot-memory works according to type A so that it is set only if the dot-lever changes its state from "unpressed" to "pressed"*. So, this mode follows the Super Keyer timing scheme but with type A instead of disabled dot-memory during the first third of a dash, with the result that the insertion of a dot by tapping the dot-lever can not fail. *And in Elecraft's "mode A" the period during which the dot-memory works according to type A is simply extended from the first third to the whole length of a dash.* So this mode gives the operator even more time to release a squeeze than the Super Keyer timing. Most Elecraft operators are very pleased with one or the other of these two modes - which is not surprising, since they are more tolerant than the true type B mode which according to my survey the vast majority of these operators initially learned. However, both modes are problematic for all those who are used to true type A or plain iambic mode.

Another example is the *PicoKeyer* which according to its specs features dot/dash-memory. The manual for the Ultra PicoKeyer (firmware V2.1, 13 January 2016) states: *"Modes A & B are simply a matter of when the keyer checks for input from the paddles. In iambic mode A, the keyer only checks for paddle inputs after the end of each dot or dash. In iambic mode B, on the other hand, the keyer will check for paddle input during each dot or dash"*. Of course this explanation is wrong, because a correctly programmed iambic keyer with dot/dash-memory checks for paddle inputs during each dot or dash in type A as well as in type B mode. But in type A it checks for a *transient* from unpressed to pressed whereas in type B it checks just for the *state* pressed. And even in plain iambic mode without dot/dash-memory it does not check after the end of each dot or dash, but after the end of each dot- or dash-*element* which contains the following space. But since the PicoKeyer indeed works according to its wrong explanation, when it is set to mode A it does not recognize paddle inputs during each dot or dash but only during the following space: key an "N" as described in the test before, if you release the dot-lever while the dash is heard the dot-memory does not work and you get a "T" instead. But if you release it a bit later during the space fol- lowing the dash, the dot-memory works and you get the "N". So the PicoKeyer in fact works neither as a true type A nor as a basic iambic keyer but instead it behaves like a strange hybrid of both. The dot/dash- memories do not work properly in ultimatic mode either, only in mode B.

Keyrama

The suffix "-rama" stems from the Ancient Greek word "οραμα" which means "wide view". In order to complement this treatise with a useful practical device, I developed the unique PIC-based multi-mode Morse code keyer *"Keyrama"* [16] which enables you to get a wide view of the different keying modes, to compare their proper logic and accurate timing with the logic and timing of other keyers, to follow the visualized action of dot/dash-memory and to find out to which extent your personal keying technique really makes use of it.

Because I could not find one single document with a proper in-depth explanation of the different twin-lever keying modes, I wrote this article to fill the gap. You don't have to know how the internal logic of your electronic keyer works to use it, but getting a handle on this subject can deepen your understanding for the process by which it generates those dits and dahs. And maybe it can excite your curiosity to try another keying mode.

Many thanks to Mort Mortimer, G2JL; Steen Wichmand, OZ8SW; Jerry Steinback, ex WAØGDS; Brian Kelley, NM7T; Kurt Nielsen from Quali-Fi and OZ7BO's son Ole Brøndum-Nielsen for their help and valuable contributions to this article. ■

References:

12.Moss, Jimmy, W5GRJ, "The WØEPV Squeeze Keyer", QST, July 1967, page 22.
13.Gensler, Harry, K8OCO, "The "Iambimatic" Concept", QST, January 1967, page 18.
14.Garrett, James M., WB4VVF, "The WB4VVF Accu-Keyer", QST, August 1973, page 19.
15.Russell, Jeffrey D., KCØQ, and Southard, Conway A., NØII, "The CMOS Super Keyer", QST, October 1981, page 11.
16.*http://cq-cq.eu/DJ5IL_rt008.pdf*
17.
18.Revisions: 7.10.2016, 20.10.2016, 7.6.2017, 3.4.2018, 18.7.2019