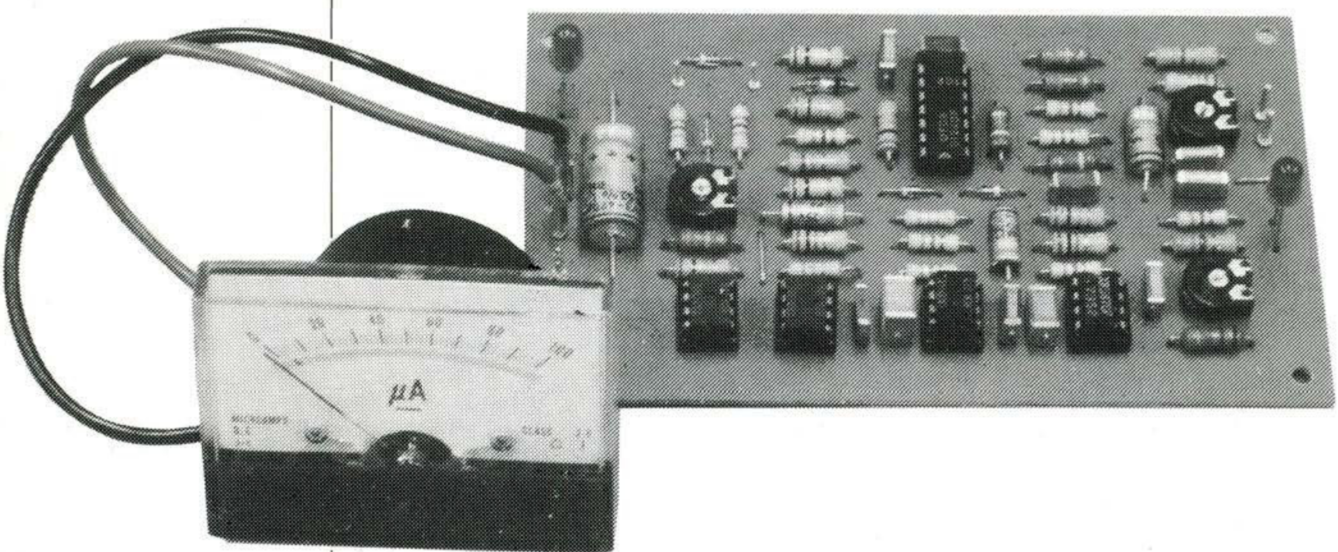
**Many radio amateurs and shortwave listeners are very interested in receiving Morse transmissions. However, the problem is understanding Morse signals, a skill that can only be learned with experience.
The alternative is to own a Junior Computer. Using a small circuit and a program, the Junior can learn to read Morse!**

Morse converter

With a Morse decoder program from R. Unterricker



copying Morse with the computer

Who can read Morse nowadays? Radio amateurs with the full (HF) licence have passed a Morse test. But this skill that was once learned with considerable effort is often forgotten, and is hardly sufficient in practice anyway. A large number of people are unable to read Morse but are interested in copying these signals. A Morse decoder gives them access to the world of dots and dashes.

This type of Morse decoder converts Morse signals to normal characters. In principle, any microcomputer can be provided with a small hardware interface and a Morse decoder program, allowing it to be used for this purpose.

This article describes a universal Morse interface and a powerful decoder program for the Junior Computer.

Decoding Morse signals

As most people know, a Morse signal consists of dots and dashes. However, the intervals between the dots and dashes are also important. Since the individual characters consist of a different number of dots and dashes, the intervals of different lengths indicate the end of a character and the end of a word.

Within a Morse character, the intervals between dots and dashes are shorter than twice the duration of a dot. Between two letters in a word the interval is longer than two, but shorter than four dots, and between two words the interval is longer than four dots.

The problem in decoding Morse is that exact timing is almost never encountered; the durations of dots and dashes can vary as well as the intervals; the ratio between these times can vary as well as the speed at which the characters are sent. There are no absolute values, and everything is relative. As an intelligent being, however, the human is able to cope with this situation as long as there is some difference between the durations of dots, dashes and intervals. This is considerably more difficult for a computer. On the other hand, it is easier to teach the computer Morse than to learn it oneself.

The many interfering signals received with the desired Morse signal on shortwave present another handicap for the computer. These can be atmospheric interference, radio

interference, interfering carriers, beating with stations on adjacent frequencies, noise and the like. A skilled Morse operator can 'copy' very weak Morse signals which are almost buried in the noise level.
Clearly, a computer Morse decoder cannot compete with these outstanding human capabilities. Nevertheless, the computer is an excellent aid for shortwave listeners, allowing most stations to be decoded. Only very weak stations or those sending extremely poor Morse will be incorrectly copied by the computer.
The Morse audio signals emitted by a receiver cannot be used by the computer. It requires an interface which converts the Morse tones to a square-wave signal which the computer can understand, and which simultaneously suppresses interference. The computer must first measure the pulse and interval durations of this square-wave signal; then, using relatively simple software routines, it must decide whether it is receiving a dash, a dot, a short, medium-length or long interval. Once the individual elements have been detected, there is no difficulty in grouping the received dots and dashes in binary-coded characters and then converting them to ASCII code. Figure 1 is a block diagram showing the structure of the microprocessor Morse decoder. A printer can be connected to the computer instead of a video interface with monitor.

The Morse interface

In principle, the function of this circuit is that of an audio-tone decoder: if a 1 kHz tone is applied to the input, a logic 1 is present at the output. If there is no signal the output goes to logic 0. An interrupted 1 kHz tone (Morse signal) results in a square-wave signal at the output, whose pulse length corresponds to the duration of the tone.
Using the beat frequency oscillator (BFO) of the shortwave receiver, the frequency of the Morse tones can be set to exactly 1 kHz. Since the Morse interface only responds to this frequency, interfering signals of a different frequency are considerably auppressed. However, the selectivity of the tone decoder is not sufficient for pulse-type interference which exhibits a wide frequency spectrum. For example, an interfering pulse appearing in a Morse signal interval could be incorrectly detected as a dot. For this reason, the circuit of the Morse interface also contains an integrator which ensures that only pulses of a certain minimum length result in a logic 1 at the output of the tone decoder. Most short interfering pulses are thus eliminated. Figure 2 explains the method of operation of the Morse interface with simplified signal diagrams.
The circuit is shown in figure 3 and the printed circuit board in figure 4.
The interface is connected to the tape

1

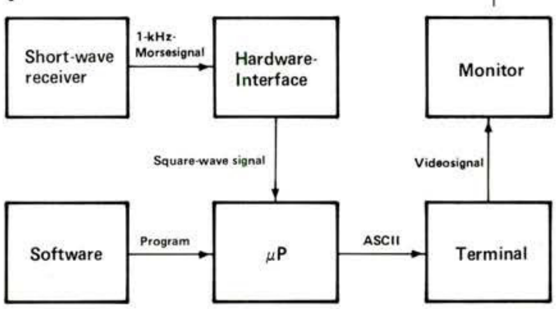


Figure 1. Block diagram of a Morse decoder using a microcomputer. The important elements are a hardware interface and decoder software.

2

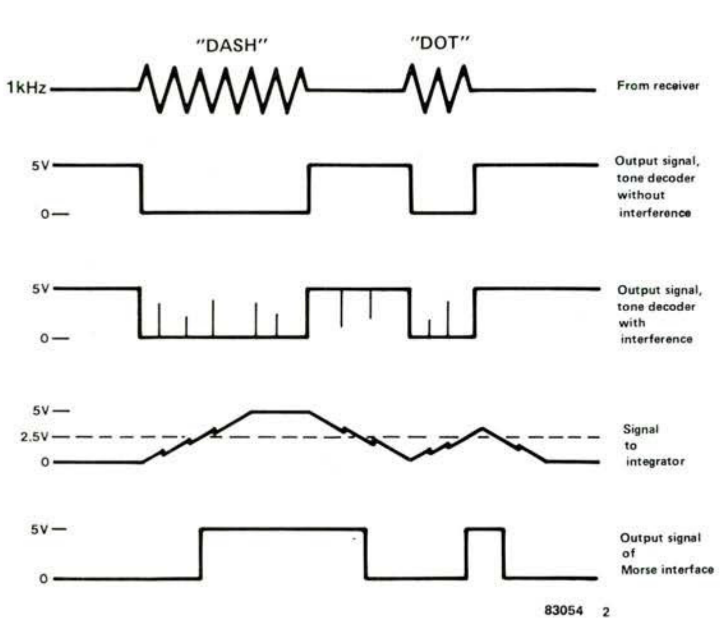


Figure 2. Simplified signal diagram for the Morse interface. A tone decoder forms a square-wave signal from the different lengths of the Morse signals. By integrating this signal with the following threshold switch (trigger), short interfering pulses are suppressed.

3

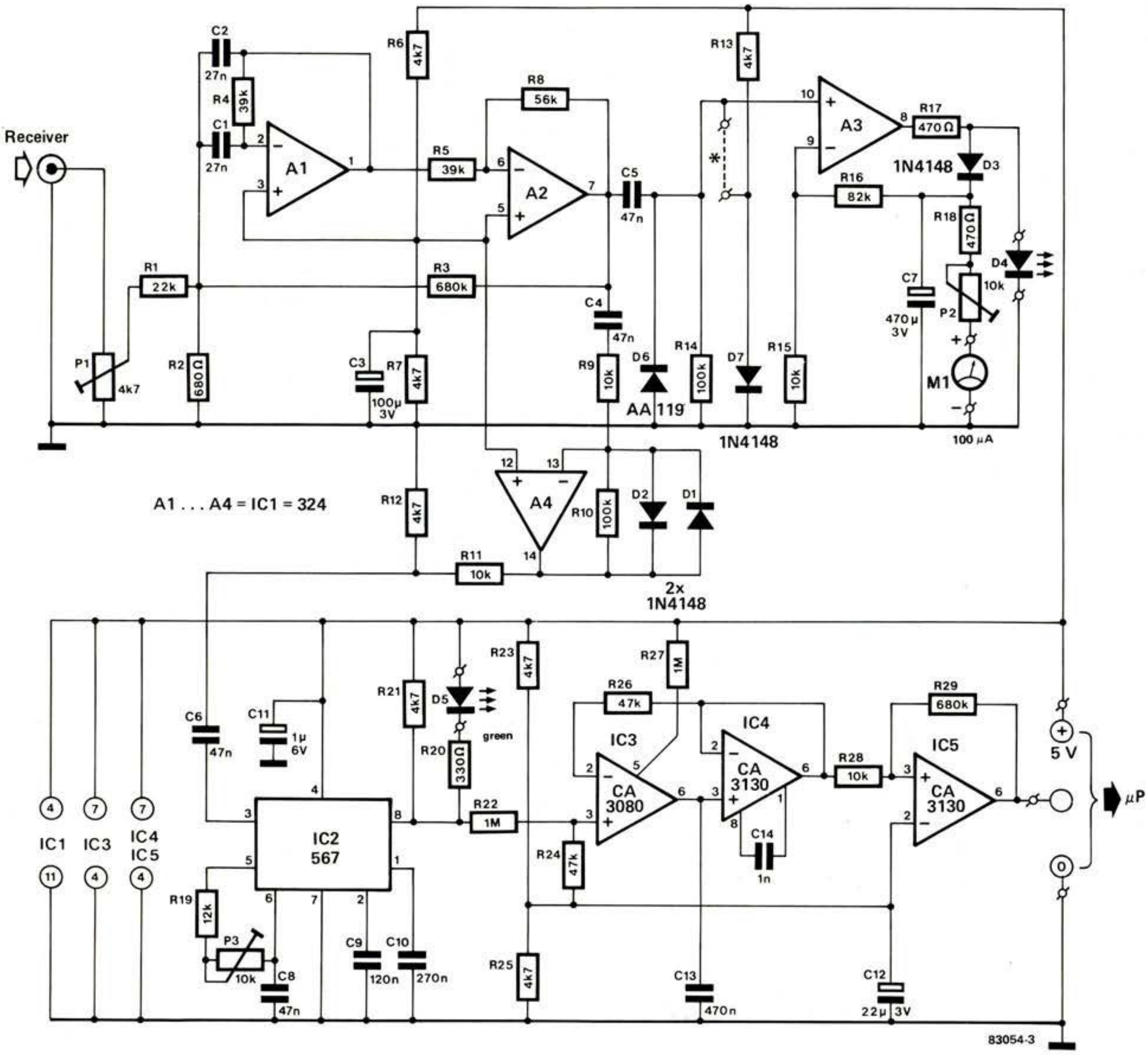


Figure 3. The Morse interface identifies the Morse tones by means of tone decoder chip IC2. A relatively involved level indication facilitates tuning of the receiver for good Morse reception.

recorder output of the receiver. A preset potentiometer at the input of the interface is used for matching the levels. A1 and A2 form an active 1 kHz filter. This is followed by amplifier A4 which has an amplification factor of 10. Diodes D1 and D2 in the feedback loop of the amplifier ensure that the output signal is limited to approximately 600 mV peak to peak value. After some attenuation at the output of A4 (R11/R12) the signal is fed via C6 to the input of the 567 tone decoder (IC2). The output of IC2, pin 8, goes to logic 0 as soon as a 1 kHz tone is applied. The green LED D5 then lights. However, the tone decoder also responds to short interfering pulses with a 0 at its output. These pulses are eliminated by the following circuit consisting of IC3 . . . IC5. A CA 3080 OTA (IC3) is configured as an integrator whose time constant is determined by the control current flowing via R27 into pin 5 and by capacitor C13. This integrator decelerates the voltage change from 0 to 1 and vice versa (see figure 2). The next operational amplifier, IC4, serves as a voltage follower in order not to load C13. IC5 is configured as a comparator with a threshold of 2.5 V. At input voltages above this value, the output (pin 6) is a logic 1. If the voltage is lower than this figure, the output presents a logic 0.
The filtered 1 kHz signal is also fed to A3, which amplifies only the positive half of the signal-waveform, and is therefore simply an 'active rectifier'. The purpose of the exercise is to drive the moving-coil meter M1 (100 μA full scale deflection) to indicate the signal strength.
Diode D7 is only required for aligning the meter; it delivers a reference voltage of 0.6 V when the jumper (drawn in dashes) is soldered in. D4, the red LED, indicates overdriving.

Aligning the Morse interface

The first step is to align the meter: insert the jumper (at D3/D7) and adjust P2 for full

4

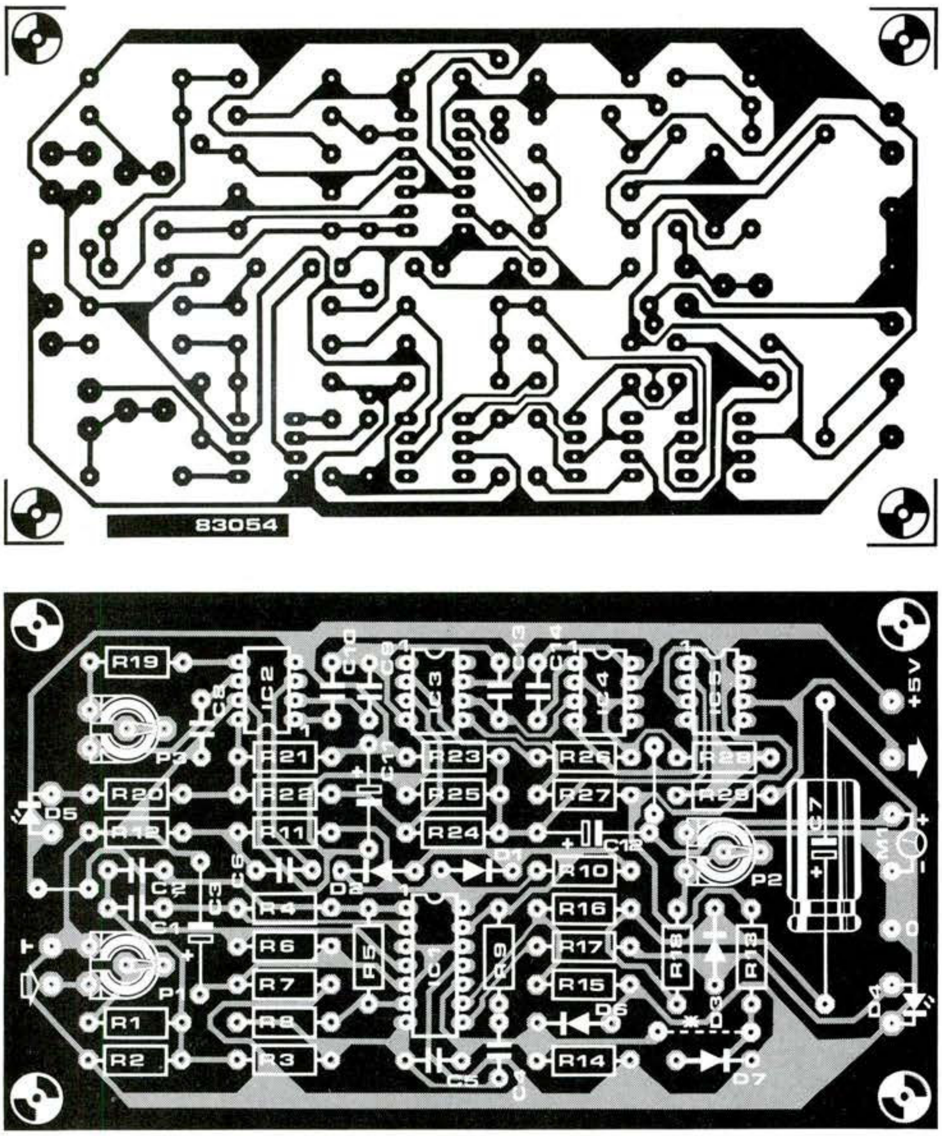


Figure 4. The printed circuit board for constructing the Morse interface; this is connected between the receiver and the computer.

scale deflection. Then remove the jumper. The receiver can now be connected. Set P1 approximately to its mid-point, and tune the receiver to maximum reading on meter M1. Remember to switch on the BFO of the receiver beforehand! In the event of overdriving, reduce the sensitivity with P1.
The tone decoder can now be aligned. Adjust P3 so that the green LED (D5) flashes at the rate of the Morse signal. The 'lock range' that can be set by adjusting P3 is relatively high. When properly adjusted, P3 should be in the middle of this lock range.

Morse decoder software

This is a program for the Junior Computer. The software contained in a 2716 EPROM is suitable for the Junior Computer with interface card extension and for the DOS Junior.
The Morse interface is connected at PB7 (6532) of the Junior Computer. For demonstrations and practising, a Morse key can also be connected at PB7 with the debounce circuit shown in figure 5.
After the start of the program, the program name 'Morse decoder' is printed out. The computer first compares the received Morse characters with each other until it detects a difference in length of at least 50 ms. Signals of a duration of less than 80 ms (interference) are ignored. As soon as the relevant time difference of more than 50 ms is detected, the Morse signals received are decoded; the first dash to be encountered serves as a time reference. With each received dash this reference is corrected, so that the decoder immediately and automatically follows any changes in sending speed.
The Morse decoder prints 64 characters per line, automatically followed by a carriage return (CR) and line feed (LF) instruction to the printer or video interface. In general, the program uses the following criteria for detecting Morse characters:

- Minimum difference between dot and

Parts list

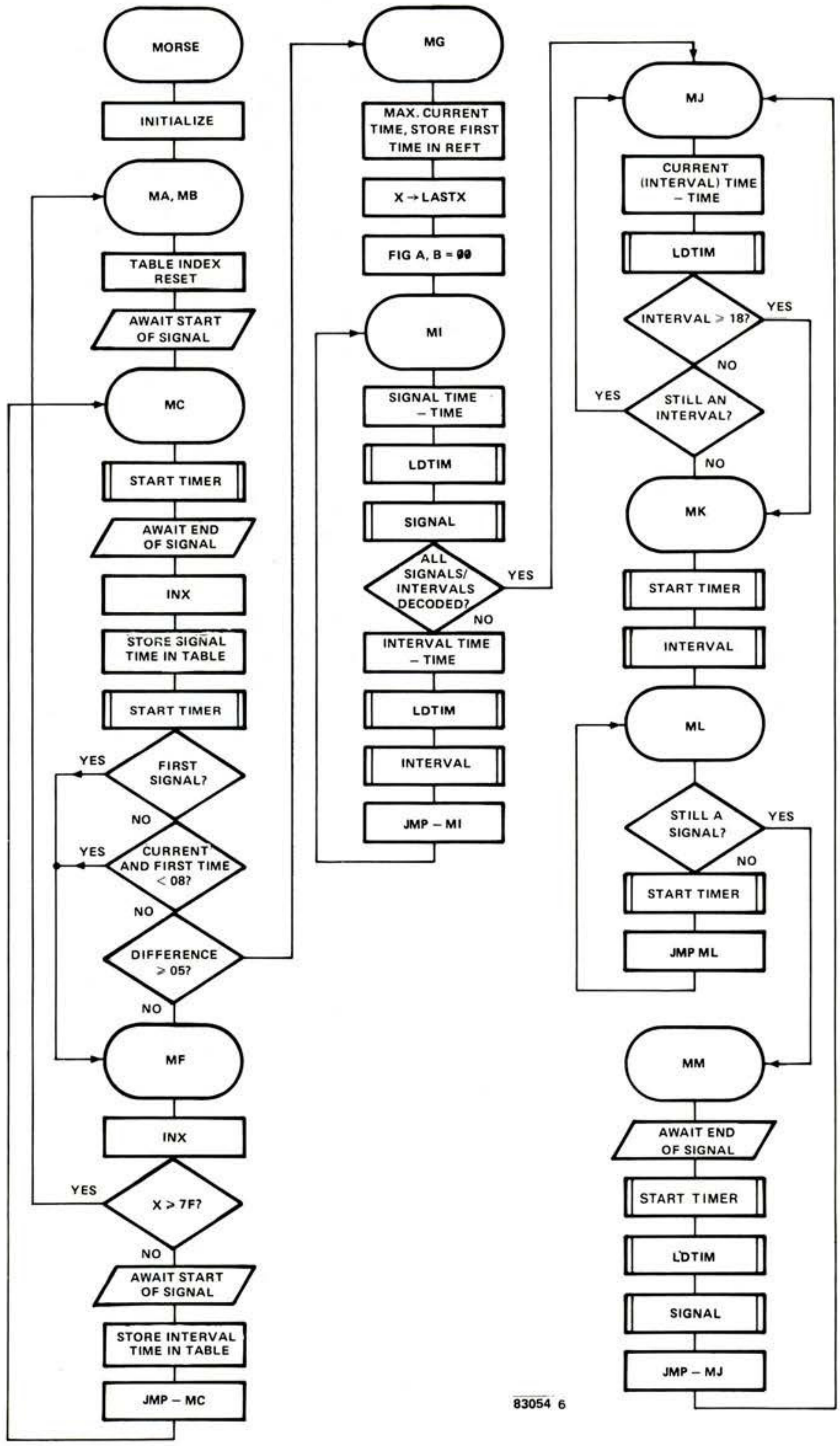
Resistors:
R1 = 22 k
R2 = 680 Ω
R3,R29 = 680 k
R4,R5 = 39 k
R6,R7,R12,R13,R21, R23,R25 = 4k7
R8 = 56 k
R9,R11,R15,R28 = 10 k
R10,R14 = 100 k
R16 = 82 k
R17,R18 = 470 Ω
R19 = 12 k
R20 = 330 Ω
R22,R27 = 1 M
R24,R26 = 47 k
P1 = 4k7 preset
P2,P3 = 10 k preset

Capacitors:
C1,C2 = 27 n
C3 = 100 µ/3 V
C4,C5,C6,C8 = 47 n
C7 = 470 µ/3 V
C9 = 120 n
C10 = 270 n
C11 = 1 µ/6 V
C12 = 22 µ/3 V
C13 = 470 n
C14 = 1 n

Semiconductors:
D1,D2,D3,D7 = 1N4148
D4 = LED red
D5 = LED green
D6 = AA 119
IC1 = LM 324
IC2 = LM 567
IC3 = CA 3080
IC4,IC5 = CA 3130

Miscellaneous:
M1, moving coil meter, 100 µA

6



5

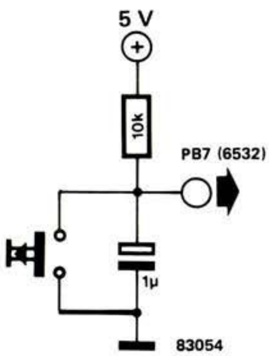


Figure 5. For practice and demonstrations a Morse key can be connected to PB7 of the Junior Computer via this debounce circuit.

dash lengths at program start
- Minimum interval for spaces
- Minimum interval for a new character (letter)
- Minimum length of a dash
- Number of characters per line

If the Morse decoder should go out of lock without relocking automatically, the program can be restarted with the NMI key. The program structure is shown in detail in the flowchart of figure 6. For reasons of space, it is not possible to provide a full listing here; we shall therefore briefly discuss the most important sections.
After establishing an initial reference time REFT, the computer is in the loop starting at label MJ. Between labels MJ and MK it awaits a Morse signal, but not longer than 18 reference time units; otherwise the last character to be received at the end of the transmission would not be decoded. This 'emergency exit' necessitates the loop under label ML, which stops the processor until the start of a new character.

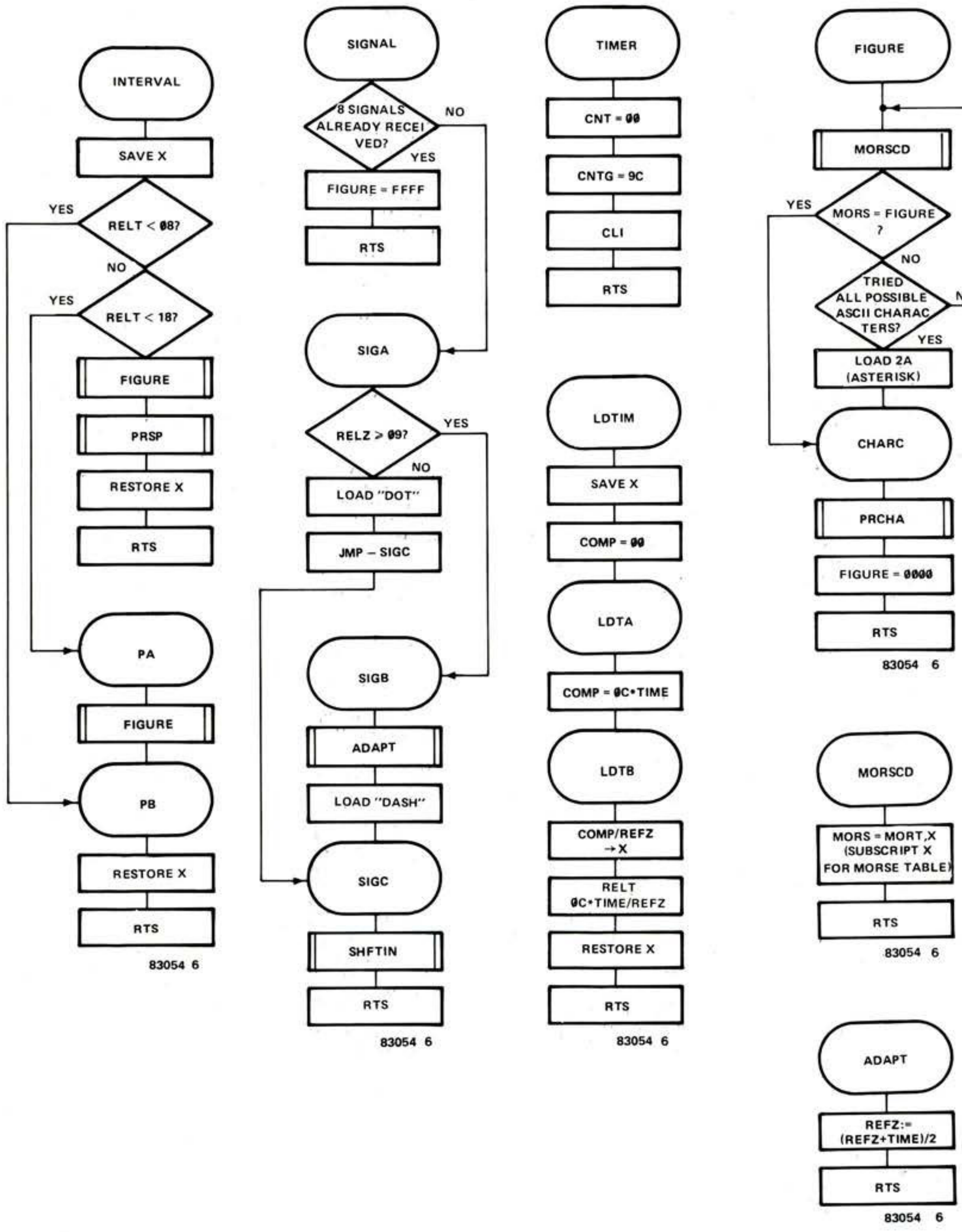


Figure 6. Flowchart of the Morse decoder with which the Junior Computer reliably converts Morse signals to plain language.

Subroutine LDTIM standardizes the time located in TIME to that in REFT: RELT = $0C TIME/REFT. Factor 0C is used to keep rounding-off errors low with this division. Subroutine FIGURE compares the Morse character located in FIG A,B with the Morse code of ASCII characters $22-$5A. In the event of agreement (identification of a Morse character) the corresponding ASCII character is printed; unidentifiable Morse characters (such as mistakes in sending) are output as an asterisk ('*'). The error signal (8 dots) of the Morse alphabet is assigned the ASCII character 23 (#).
Subroutine SHFTIN shifts a recognized dot or dash into the cells of FIG A and FIG B. As can be seen in figure 7, 00 is a space, 01 is a dot, 10 is a dash and 11 is an error.

Instructions for using the program

The program requires a memory range of 4000 to 7FFF (RAM). A (dynamic) 16 k RAM card on the Junior bus is sufficient.

Table 1. Start addresses of the copying routines

Junior Computer version	Copying routine start add.	Copies from address	to address
expanded	ØE56	Ø8ØØ	4ØØØ
DOS	EB2D	E8ØØ	4ØØØ

Table 2. Modifications to DOS Junior/Morse program

Address	Data
4238	A3
4239	FE
426D	A3
426E	FE
4284	A3
4285	FE
428C	A3
428D	FE

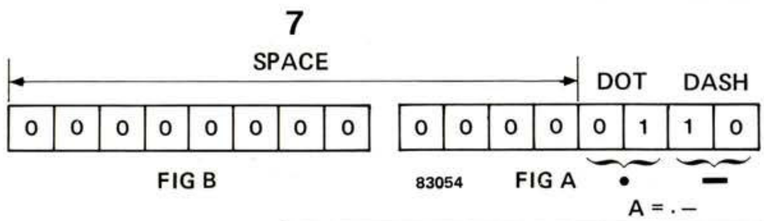
Table 3. Modifications to extended Junior/Morse program

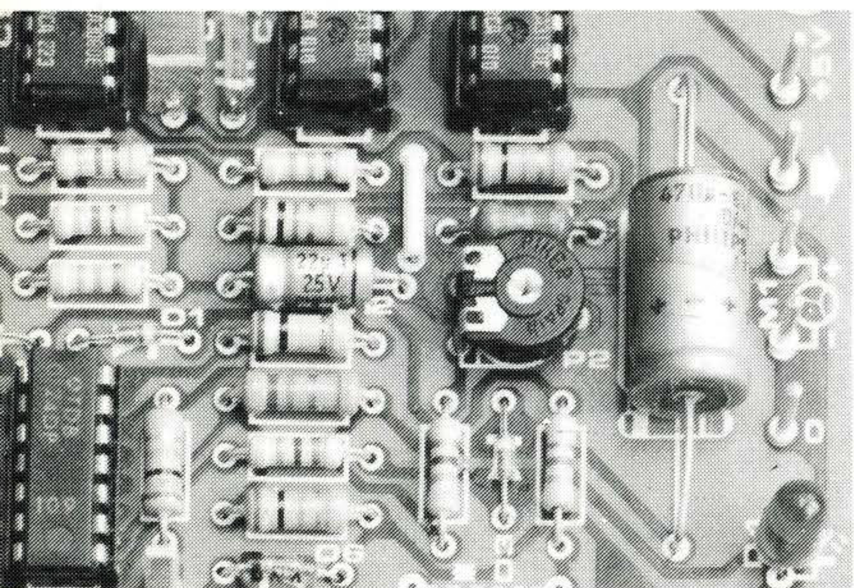
Address	Data	Address	Data
4013	1A	4238	34
4018	1A	4239	13
401B	1A	4245. . .	EA, EA, EA
4022	1A	4251	1A
402C	1A	4268	1A
4031	1A	426D	34
4039	1A	426E	13
403C	1A	4280. . .	EA, EA, EA
4048	1A	4284	34
404D	1A	4285	13
40E1	1A	4288. . .	EA, EA, EA
40F8	1A	428C	34
414F	1A	428D	13
41F5. . .	EA, EA, EA		
4234. . .	EA, EA, EA		

Table 4. Hexdump listing of the Morsedecoder-program.

	Ø	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8ØØ	4C	11	4Ø	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ
81Ø	ØØ	AD	82	FA	29	DF	8D	82	FA	AD	83	FA	Ø9	2Ø	29	7F
82Ø	8D	83	FA	A9	ØØ	8D	ØF	4Ø	A9	3F	8D	7A	FA	A9	42	8D
83Ø	7B	FA	2Ø	13	BC	A9	81	8D	F7	FA	2C	D5	FA	1Ø	FB	AØ
84Ø	ØØ	2Ø	2D	42	A9	4C	8D	7E	FA	A9	42	8D	7F	FA	A2	FF
85Ø	2Ø	22	41	2Ø	46	41	2Ø	34	41	E8	AD	Ø3	4Ø	9D	2C	43
86Ø	2Ø	46	41	8A	FØ	2Ø	AD	2C	43	C9	Ø8	1Ø	Ø7	BD	2C	43
87Ø	C9	Ø8	3Ø	12	38	BD	2C	43	ED	2C	43	BØ	Ø5	49	FF	18
88Ø	69	Ø1	C9	Ø5	1Ø	11	E8	EØ	7F	1Ø	C3	2Ø	22	41	AD	Ø3
89Ø	4Ø	9D	2C	43	4C	53	4Ø	BD	2C	43	CD	2C	43	1Ø	Ø3	AD
8AØ	2C	43	8D	Ø4	4Ø	8E	Ø5	4Ø	A2	ØØ	8E	Ø6	4Ø	8E	Ø7	4Ø
8BØ	BD	2C	43	8D	Ø8	4Ø	2Ø	52	41	2Ø	B2	41	EC	Ø5	4Ø	FØ
8CØ	11	E8	BD	2C	43	8D	Ø8	4Ø	2Ø	52	41	2Ø	91	41	E8	4C
8DØ	BØ	4Ø	AD	Ø3	4Ø	8D	Ø8	4Ø	2Ø	52	41	C9	18	1Ø	11	AD
8EØ	82	FA	29	8Ø	DØ	EC	2Ø	5C	42	DØ	E7	2Ø	5C	42	DØ	E2
8FØ	2Ø	46	41	2Ø	91	41	AD	82	FA	29	8Ø	FØ	Ø6	2Ø	46	41
9ØØ	4C	F6	4Ø	2Ø	5C	42	DØ	EE	2Ø	5C	42	DØ	E9	2Ø	34	41
91Ø	AD	Ø3	4Ø	8D	Ø8	4Ø	2Ø	46	41	2Ø	52	41	2Ø	B2	41	4C
92Ø	D2	4Ø	2Ø	66	42	DØ	FB	2Ø	5C	42	DØ	F6	2Ø	5C	42	DØ
93Ø	F1	4C	66	42	2Ø	66	42	FØ	FB	2Ø	5C	42	FØ	F6	2Ø	5C
94Ø	42	FØ	F1	4C	66	42	A9	ØØ	8D	Ø3	4Ø	A9	9C	8D	FE	FA
95Ø	58	6Ø	8E	Ø9	4Ø	A2	ØØ	8E	ØA	4Ø	8E	ØB	4Ø	A2	ØC	18
96Ø	AD	ØA	4Ø	6D	Ø8	4Ø	8D	ØA	4Ø	AD	ØB	4Ø	69	ØØ	8D	ØB
97Ø	4Ø	CA	DØ	EB	38	AD	ØA	4Ø	ED	Ø4	4Ø	8D	ØA	4Ø	AD	ØB
98Ø	4Ø	E9	ØØ	8D	ØB	4Ø	E8	BØ	EC	8A	8E	ØC	4Ø	AE	Ø9	4Ø
99Ø	6Ø	8E	Ø9	4Ø	AD	ØC	4Ø	C9	ØA	3Ø	13	C9	18	3Ø	ØC	2Ø
9AØ	D7	41	78	2Ø	43	42	58	AE	Ø9	4Ø	6Ø	2Ø	D7	41	AE	Ø9
9BØ	4Ø	6Ø	AD	Ø7	4Ø	29	CØ	FØ	Ø9	A9	FF	8D	Ø6	4Ø	8D	Ø7
9CØ	4Ø	6Ø	AD	ØC	4Ø	C9	Ø9	1Ø	Ø5	A9	Ø2	4C	D3	41	2Ø	12
9DØ	42	A9	Ø1	2Ø	1E	42	6Ø	A2	22	2Ø	Ø5	42	AD	ØD	4Ø	CD
9EØ	Ø6	4Ø	DØ	Ø8	AD	ØE	4Ø	CD	Ø7	4Ø	FØ	Ø7	E8	EØ	5B	3Ø
9FØ	E8	A2	2A	8A	78	8D	63	23	2Ø	6C	42	58	A2	ØØ	8E	Ø6
AØØ	4Ø	8E	Ø7	4Ø	6Ø	BD	6F	42	8D	ØD	4Ø	BD	AF	42	8D	ØE
A1Ø	4Ø	6Ø	18	AD	Ø4	4Ø	6D	Ø8	4Ø	4A	8D	Ø4	4Ø	6Ø	4A	2E
A2Ø	Ø6	4Ø	2E	Ø7	4Ø	4A	2E	Ø6	4Ø	2E	Ø7	4Ø	6Ø	B9	ØF	43
A3Ø	C9	ØØ	FØ	ØA	8D	63	23	2Ø	43	23	C8	4C	2D	42	6Ø	78
A4Ø	4C	11	4Ø	A9	2Ø	8D	63	23	2Ø	6C	42	6Ø	48	A9	9C	8D
A5Ø	FE	FA	EE	Ø3	4Ø	DØ	Ø3	CE	Ø3	4Ø	68	4Ø	A9	7F	8D	1Ø
A6Ø	4Ø	CE	1Ø	4Ø	DØ	FB	AD	82	FA	29	8Ø	6Ø	2Ø	43	23	EE
A7Ø	ØF	4Ø	A9	3F	CD	ØF	4Ø	DØ	15	A9	ØØ	8D	ØF	4Ø	A9	ØD
A8Ø	8D	63	23	2Ø	43	23	A9	ØA	8D	63	23	2Ø	43	23	6Ø	ØØ
A9Ø	ØØ	59	55	ØØ	ØØ	ØØ	A9	69	A6	ØØ	ØØ	5A	56	66	59	AA
AAØ	AA	6A	5A	56	55	55	95	A5	A9	95	99	ØØ	56	ØØ	A5	ØØ
ABØ	Ø6	95	99	25	Ø1	59	29	55	Ø5	6A	26	65	ØA	Ø9	2A	69
ACØ	A6	19	15	Ø2	16	56	1A	96	9A	A5	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ
ADØ	ØØ	Ø6	55	ØØ	ØØ	ØØ	Ø6	Ø2	Ø9	ØØ	ØØ	ØA	Ø9	Ø6	Ø2	Ø2
AEØ	Ø1	Ø1	Ø1	Ø1	Ø1	Ø2	Ø2	Ø2	Ø2	ØA	Ø9	ØØ	Ø2	ØØ	Ø5	ØØ
AFØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ
BØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØD
B1Ø	ØA	ØA	ØA	4D	4F	52	53	45	2Ø	44	45	43	4F	44	45	52
B2Ø	ØD	ØA	ØA	52	45	41	44	59	ØD	ØA	ØA	ØØ	ØØ	A9	ØØ	A2
B3Ø	E8	85	ØØ	86	Ø1	A9	ØØ	A2	4Ø	85	Ø2	86	Ø3	2Ø	43	EB
B4Ø	4C	ØØ	FC	A2	Ø4	AØ	ØØ	B1	ØØ	91	Ø2	88	DØ	F9	E6	Ø1
B5Ø	E6	Ø3	CA	DØ	FØ	6Ø	A9	ØØ	A2	Ø8	85	ØØ	86	Ø1	A9	ØØ
B6Ø	A2	4Ø	85	Ø2	86	Ø3	2Ø	6C	E8	4C	1D	1C	A2	Ø4	AØ	ØØ
B7Ø	B1	ØØ	91	Ø2	88	DØ	F9	E6	Ø1	E6	Ø3	CA	DØ	FØ	6Ø	

Figure 7. The subroutine shifts a recognized dot or dash into cells FIG A and FIG B.





The start address is 4ØØØ.
Since the DOS Junior has a different memory structure to that of the expanded Junior, two versions of the Morse program are accommodated in one EPROM. The EPROM is plugged into the socket for IC4 on the Junior extension card.
With the extended Junior it is located in address range Ø8ØØ to ØFFF. With the DOS Junior, on the other hand, it is located in address range E8ØØ to EFFF. Before the program can be started it must be copied from the EPROM into the RAM. The necessary copying routines are already contained in the EPROM. The start addresses of the copying routines are specified in table 1.
Once the program has been copied from the EPROM into the RAM, some bytes must be changed manually by typing in the specified bytes at the addresses mentioned on table 2 or 3.
After this operation the program can be started; it can also be written from the RAM onto a cassette or floppy disk (with the DOS Junior) to facilitate future reuse.
Those readers wishing to program a type 2716 EPROM with the Morse program themselves will find the hex dump listing in table 4. The ready programmed 2716 is available from Technomatic Limited.