Oric Tape Format (tap file format)
By Chloé Avrillon (aka Ladywasky, formerly waskol)

Multi-byte values are in big-endian convention : Value or Address=byte1*256+byte2

Arrays are zero based in basic !

	Bloc (explanation)	Number of bytes	Possible values	Comment
Typical Bloc BASIC or memory bloc (data or machine code)	Synchronisation (N+1) bytes	N ≥ 4 Typically : Fast : 16bytes Slow : 8bytes	$16	synchronisation bytes
		1	$24	end of synchronisation
	Header 9 bytes	2	varies (unused)	reserved bytes
		1	$00	BASIC
			$80	Machine code or memory bloc
		1	$00	Autorun OFF
			other value ($C7)	Autorun ON
		1	EndAddress1 (high)	Memory locations
		1	EndAddress2 (low)	
		1	StartAddress1 (high)	Address=byte1*256+byte2
		1	StartAddress2 (low)	**SizeOfData**=EnAdrr-StartAdrr+1
		1	varies (unused)	
	File name 15 bytes max + $0	15 max	ASCII values	Name
		1	$00	end of name
	File data	**SizeOfData**	varies	succession of bytes

Note : no ending bytein oricfiles.

	Bloc (explanation)	Number of bytes	Possible values	Comment
Array of integers (STORE/RECALL)	Synchronisation (N+1) bytes	N ≥ 4 Typically : Fast : 16bytes Slow : 8bytes	$16	synchronisation bytes
		1	$24	end of synchronisation
	Header	1	$80	Flag integer/real (used by v1.1 ROM only)
		1	$00	Flag string
			$40	Array
		1	$00	Autorun OFF
			other value ($C7)	Autorun ON
		1	StartAddress1 (high)	Address of array when saved.
		1	StartAddress2 (low)	Unused. $FFFF for bi-dim.
		1	SizeOfData1 (high)	SizeOfArray=byte1*256+byte2
		1	SizeOfData2 (low)	SizeofData=SizeOfArray(*)
		1	$FF	unused
	File name 15 bytes max + $0	15 max	ASCII values	Name
		1	$00	end of name
	File data	**(SizeOfData div 2) "integers"**	varies	succession of integers (2 bytes)

Note : no ending indicator, The values are stored sequentially without dimension, nor variable name

(*) Not accurate : it's seems to be a bug in Oric rom, the Size of the array is wrong. In reality, SizeofData=SizeOfArray-offset
The offset value is typically 6 for a single dimensional array, 8 for a bi-dimensional array (and it is not always true).
Either the datalength provided is false, either are missing the array descriptors in the data (dimensions for instance)

	Bloc (explanation)	Number of bytes	Possible values	Comment
Array of reals (STORE/RECALL)	Synchronisation (N+1) bytes	N ≥ 4 Typically : Fast : 16bytes Slow : 8bytes	$16	synchronisation bytes
		1	$24	end of synchronisation
	Header	1	$00	Flag integer/real (used by v1.1 ROM only)
		1	$00	Flag string
			$40	Array
		1	$00	Autorun OFF
			other value ($C7)	Autorun ON
		1	StartAddress1 (high)	Address of array when saved.
		1	StartAddress2 (low)	Unused. $FFFF for bi-dim.
		1	SizeOfData1 (high)	SizeOfArray=byte1*256+byte2
		1	SizeOfData2 (low)	SizeofData=SizeOfArray(*)
		1	$FF	unused
	File name 15 bytes max + $0	15 max	ASCII values	Name
		1	$00	end of name
	File data	**SizeOfData x 5bytes**	varies	succession of reals (5 bytes)

Note : no ending indicator, The values are stored sequentially without dimension, nor variable name

(*) same as for array of integers

	Bloc (explanation)	Number of bytes	Possible values	Comment
Array of strings (STORE/RECALL)	Synchronisation (N+1) bytes	N ≥ 4 Typically : Fast : 16bytes Slow : 8bytes	$16	synchronisation bytes
		1	$24	end of synchronisation
	Header	1	$00	Flag integer/real (used by v1.1 ROM only)
		1	$FF	Flag string
			$40	Array
		1	$00	Autorun OFF
			other value ($C7)	Autorun ON
		1 1	$FF $FF	unused
		1 1	SizeOfData1 (high) SizeOfData2 (low)	Value=byte1*256+byte2 SizeofData=SizeOfArray(*)
		1	$FF	unused
	File name 15 bytes max + $0	15 max 1	ASCII values $00	Name end of name
	File data	**SizeOfData+strings**	varies	succession of bytes
	n=SizeOfData/3 n of theses --> (n descriptors of 1 string)	**1** **2**	varies varies	String length Address of the string (useless !)
	Strings	**len(0)+len(1)+..+len(n-1)**	varies	N Strings concatenated

Note : no ending indicator, The string values are stored sequentially without dimension, nor variable name, nor separators

(*) same as for array of integers