

offset dec hex		type name in struct def	value	meaning	note	
tof = TDC bin number indicating the time-of-flight of a detected ion						
0	0	char[4] fileID	'C','R','D','\0'	file ID	Chili/chicago Raw Data format/file	CRDHEADER (88 bytes)
...	
4	4	char[20] startDateTime	"YYYY:MM:DD hh:mm:ss"		permanent time stamp in human readable format	
...	date and time from start of measurement, 24h format, eg: "2014-01-15 14:36:51"	
24	18	uint16 minVer	0	minor file type revision	could be read/written as one uint32	
...	
26	1A	uint16 majVer	1	major file type version		
...	
28	1C	uint32 sizeOfHeaders	88	size of all header info [bytes]	= sizeof(CRDHEADER) for now, could be used later for old software to try reading in newer file versions that have additional headers but known tof format, ie after absolute file offset sizeOfHeaders either a POSTABLE or SCANDATA follows	
...	
32	20	uint32 shotPattern	0 or 32	pattern of shots within scan	support 0 and 32 from beginning, then 1 and 128, rest maybe later 0: no scan, implies xDim = yDim = nofScans = 1, shotsPerPixel = nofShots 1: user defined via translation table #shot -> (xPos, yPos) after main header, TBD 32: by line, starting upper left (left to right) 33: by line, starting lower left (left to right) 34: by line, starting upper right (right to left) 35: by line, starting lower right (right to left) ... 48: meandering by line, starting upper left (left to right) 49: meandering by line, starting lower left (left to right) ... 64: by column, up to down, starting upper left ... 128: pseudo random: 2x2 recursion, ul,ur,ll,lr sequence 129: pseudo random: 2x2 recursion, ul,lr,ll,ur sequence ...	
...	
36	24	uint32 tofFormat	1	how time-of-flight data is stored in each scan	for now, only a fast sequential mode, later we look into compression 0: no raw data in this file (just for book-keeping purposes) 1: no scan guards, first # tofs as uint32, then each tof as uint32	
...	
40	28	uint32 polarity	0 or 1	0 = positive, 1 = negative	important for mass calibration. note: most meta data / exp. parameters are to be found in a human readable text file CSV/TBD	
...	
44	2C	uint32 binLength	100	length of time bin [ps]	important for creating a time axis and dead time correction. note: most meta data / exp. parameters are to be found in a human readable text file CSV/TBD	
...	
48	30	uint32 binStart	(user)	first time bin used [bin number]	essential for processing tofs: allocating just enough RAM to create spectrum	
...	
52	34	uint32 binEnd	(user)	last time bin used [bin number]	essential for processing tofs: allocating just enough RAM to create spectrum	
...	
56	38	uint32 xDim	(user)	x-dim [pixel] of (master) raster	shotPattern 128,129 require xDim = yDim = 2^n, for shotPattern 1: size of master raster the user defined pattern is part of	
...	if not given, x-dim shall be assumed, for shotPattern 1: size of master raster which encompasses user defined pattern	
60	3C	uint32 yDim	(user)	y-dim [pixel] of (master) raster	probably = 1 for raster measurements, will be = nofShots for shotPattern 0	
...	
64	40	uint32 shotsPerPixel	(user)	shots per pixel per scan		
...	usually = xDim*yDim, unless shotPattern 1	
68	44	uint32 pixelPerScan	(user)	pixel per scan		
...	should be used to check for data integrity only	
72	48	uint32 nofScans	(user)	total # scans		
...	should be used to check for data integrity only, discrepancy with pixelPerScan*shotsPerPixel*nofScans will indicate incomplete last scan	
76	4C	uint32 nofShots	(user)	total # shots		
...	estimate of the time difference between TDC time 0 and the time SI are accelerated (if not known, make it 0.0)	
80	50	double deltaT	(user)	time diff TDC to SI accel. [s]		
...		
88	58					(POS TABLE)

not supported from beginning. would only be here if shotPattern = 1
probable size = pixelPerScan*2 (ie unit16 xPos)*2 (ie unit16 yPos) bytes

shotPattern = 34-byte bin number

88	58	uint32	N(1)	# tofs in shot #1		SCANDATA
...	
92	5C	uint32		tof 1	TBD: highest bit set = TDC overflow?	
...	
96	60	uint32		tof 2		
...	
100	64	uint32		...		
...	
40+4*N(1)		uint32		tof N(1)		
...	
40+4*N(1)+4		uint32	N(2)	# tofs in shot #2		
...	

and so on covering all shots

EOF-4		char[4]	'O','K','I','\0'	indicate successful export	also potentially helpful to 'borrow' a byte when reading in compressed data, eg read 3-byte tofs as uint32, then && x00FFFFFF	End Tag
...	
EOF						