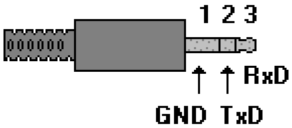
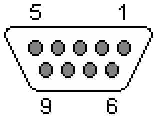
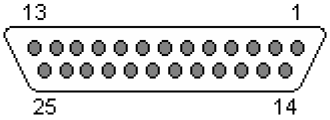
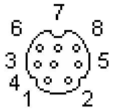
Serial Cable

The serial RS-232 interface of the Kodak DC20 and DC25 needs only 2 signal lines (RxD and TxD) and Ground.

to Camera		**to PC (e.g. COM 1)**	**to PC (e.g. COM 2)**	**to Mac**
				
1	wire to	**5** (GND)	**7** (GND)	**4** (GND) and **8** (RxD+)
2	wire to	**3** (TxD)	**2** (TxD)	**3** (TxD-)
3	wire to	**2** (RxD)	**3** (RxD)	**5** (RxD-)

Serial Protocol

Via the RS232 interface you could control the camera and download the image data. You know that and you are familiar with the Kodak software. But if you don't like Windows and you haven't got a Mac, you don't get any help from Kodak.

With the information about the serial communication protocol it would be easy to write a driver for LINUX, OS/2 or any other OS. It would be no problem to control the DC20 from your palmtop or your PDA too. You could write your own program to make slow motion photos. If you like, you could use the camera for stand alone observation with automatic transmission of the pictures via modem. And so on, and so on ...

And now you'll get the information you need.

Note: Special info about the DC25 available.

Initialization:

After power on the DC20 waits for serial data with the parameters 9600 baud, 8 data bits, 1 stop bit, even parity. You must send an init string. With this string you can set the baud rate to a new value. The camera sends a response byte and you know the camera can hear you.

init string (hex)	description
41 00 96 00 00 00 00 1A	new baud rate is 9600
41 00 19 20 00 00 00 1A	new baud rate is 19200
41 00 38 40 00 00 00 1A	new baud rate is 38400
41 00 57 60 00 00 00 1A	new baud rate is 57600
41 00 11 52 00 00 00 1A	new baud rate is 115200

The init string has the same structure like the other commands. Each command consists of 8 bytes. The first byte is the command byte the last byte is ever 0x1A (Ctrl-Z). The other 6 bytes are mostly null.

If you change the baud rate, the response for the change command is sent with the old baud rate. But after that the camera understands only the new baud rate.

The response from the camera on every command is 0xD1.

Status Information:

The next most important command is reading the camera status information. After initialization you send the

command string: 7F 00 00 00 00 00 00 1A (hex).

First you'll get the normal response byte D1. After that the camera sends 257 status bytes. The first 256 bytes are the information and the byte number 257 is the checksum. The checksum is the XOR of all 256 info bytes.

Now you must send 0xD2 to acknowledge and the communication ends with a null byte sending by the camera.

In the 256 information bytes I'm sure about the following meanings:

byte	description
2	model (DC20: 0x20; DC25: 0x25)
10	number of taken pictures
12	number of remaining pictures
24	0 if high resolution, 1 if low res

30	1 if battery down, else 0

Take A Photo:

command string: 77 00 00 00 00 00 00 1A (hex)

response: D1 and a zero byte if camera is ready

Note: The byte D1 is sent in the moment when the shutter opens. The byte is sent at 9600 baud (independent of the chosen communication baud rate). The byte will be sent also if you push the trigger at the camera. This could be used for synchronization of an external flash device. The delay between sending the command string (or pushing the trigger) and getting the response (shutter opens) is about half a second.

Erase Memory:

command string: 7A 00 00 00 00 00 00 1A (hex)

response: D1 and a zero byte if camera is ready

Change Resolution:

change to high-res: 71 00 00 00 00 00 00 1A (hex)

response: D1

change to low-res: 71 00 01 00 00 00 00 1A (hex)

response: D1

If memory contains pictures, it doesn't work.

Download Thumbnails:

command string: 56 00 00 01 00 00 00 1A (hex)
Byte No. 4 is the picture number you want to receive.

response: D1

After that the camera sends five data blocks. Each data block consists of 1024 bytes + 1 byte for checksum. The checksum is the XOR of all 1024 data bytes. After each received block you must send a D2 byte as acknowledgment.

The communication ends with a null byte sending by the camera.

Now you have got 5120 data bytes. Read the first 4800 bytes as a byte array of 60 rows and 80 columns and each byte in it as a pixel scaled in 256 shades of grey (from 0 for black to 255 for white).

Download Pictures:

command string: 51 00 00 01 00 00 00 1A (hex)
Byte No. 4 is the picture number you want to receive.

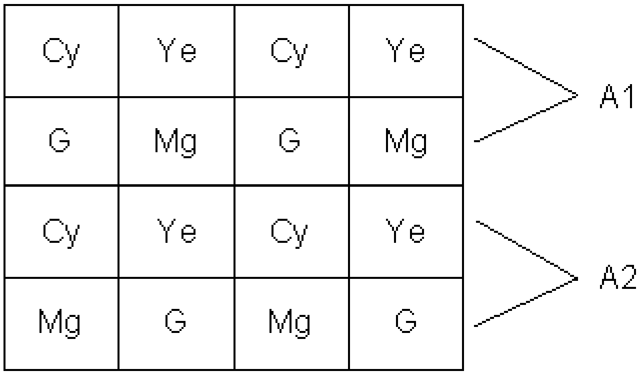
response: D1

After that the camera sends 122 data blocks (if in high-res mode) or 61 data blocks (if in low-res mode). Each data block consists of 1024 bytes + 1 byte for checksum. The checksum is the XOR of all 1024 data bytes. After each received block you must send a D2 byte as acknowledgment.

The communication ends with a null byte sending by the camera.

Image Format:

The CCD in the camera most likely consists of 501 (H) * 484 (V) usable pixels. The combination with a complementary color mosaic filter (Yellow, Cyan, Magenta, Green) produces the following pixel pattern:



Cy	Ye	Cy	Ye
G	Mg	G	Mg
Cy	Ye	Cy	Ye
Mg	G	Mg	G

Each two adjacent rows are added and the new rows (A1, A2, ...) contain the image information stored in the camera memory and sent to the computer. These new rows consist of pixels with [G+Cy], [Mg+Ye], [G+Cy], [Mg+Ye], ... color information (in odd rows) or with [Mg+Cy], [G+Ye], [Mg+Cy], [G+Ye], ... color information (in even rows).

For a high-res picture the camera sends 122*1024= 124.928 bytes. The first 512 bytes are a header. The rest are 243 rows with 512 pixels. The camera stores and sends only the raw data from the CCD. There is no image processing or compression inside the camera. Not all of the raw data pixels are usable (see table below).

mode	file size	header	raw data	left margin	right margin	top margin	bottom margin
high res	124.928	512	512 * 243	1 column	10 columns	-	1 row
low res	62.464	256	256 * 243	1 column	5 columns	-	1 row

See the following table for image header details:

offset	type	description
0	BYTE	always 1
1	BYTE	0x20 for DC20
2	UINT16	picture number
4	BYTE	0 for high res, 1 for low res
5	BYTE	always 0
6	UINT32	file size in byte: 0x1E800 (high) or 0xF400 (low)
10..15	BYTE	always 0
16	INT8	EXP1: value depends on lighting conditions
17	INT8	EXP2: value depends on lighting conditions
18..27	BYTE	always 0
28..29	BYTE	always FF
30..33	BYTE	always 0
34	INT8	EXP3: value depends on lighting conditions
35	INT8	EXP4: value depends on lighting conditions
36..41	BYTE	always { 0xB6, 0xC8, 0xD6, 0xAB, 0x6E}
42..END		unknown

Note: UINTxx means unsigned integer (little-endian) with xx bits

See dc20_exp.htm for more information about the exposure data EXP1..4.

The easiest way to convert the raw data to grey scale shows the following routine:

```
for (row= 0; row < 243; row++)
  for (col= 0; col < 512; col+=2)
    grey_pic[row][col/2]= (ccd[row][col] + ccd[row][col+1])/2;
```

You will get a 256*243 pixel image. If you stretch it horizontally to 128% it looks fine. It's necessary because the CCD cells aren't quadratic.

To extract the color information as RGB you can use the program cmttoppm.c written by YOSHIDA Hideki. For more information see http://www.yk.rim.or.jp/~hideki/es1000/index-e.html. His Chinon ES-1000 seems to use the same hardware like the Kodak DC20.

Homepage: http://www.planet-interkom.de/oliver.hartmann