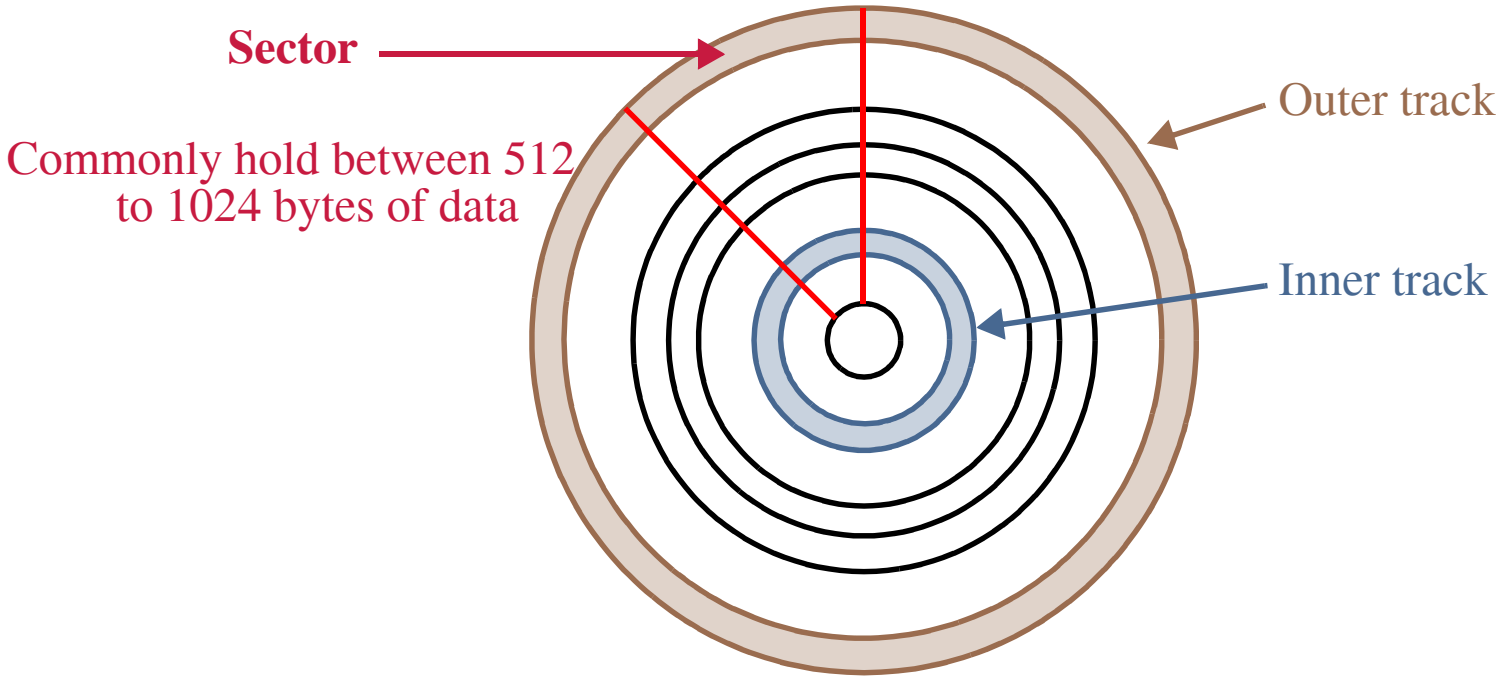
## *Disk Memory Systems*

Magnetic and optical:

- Floppy disks
- Hard disks
- CD-ROMs and WORMs (write once/read mostly)
- DVD

Floppy:



## *Floppy*

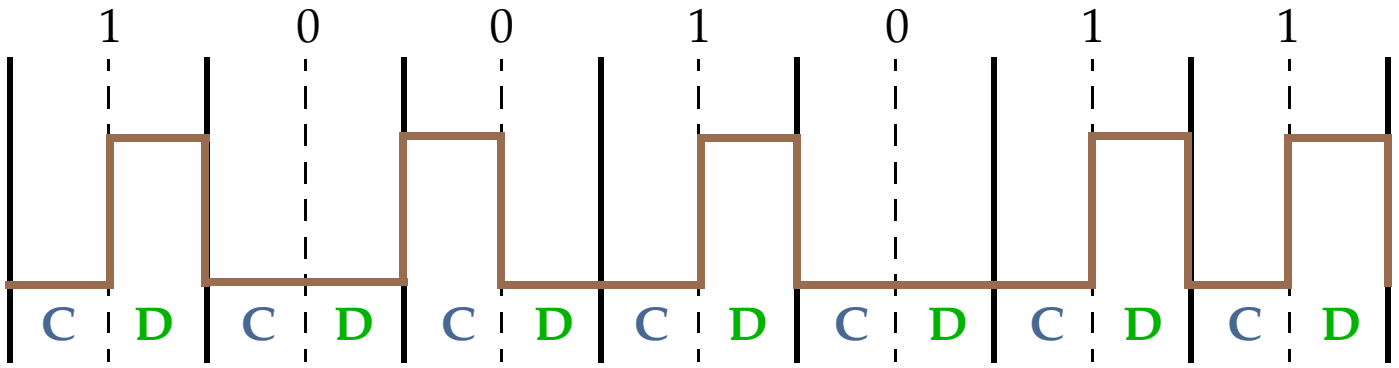
Older 5 and 1/4 flexible floppies spin at 300 RPM, have 40 tracks with 9 sectors/track and two sides.

Capacity = 40 *X* 2 *X* 9 *X* 512 = 368,640 or ~360K bytes of information.

Newer ones are high-density with 80 tracks and 15 sector/track for 1.2 MB.

Heads actually contact the disk surface, leading to wear out.

The recording format called **MFM** (modified frequency modulation) used to write double density format.



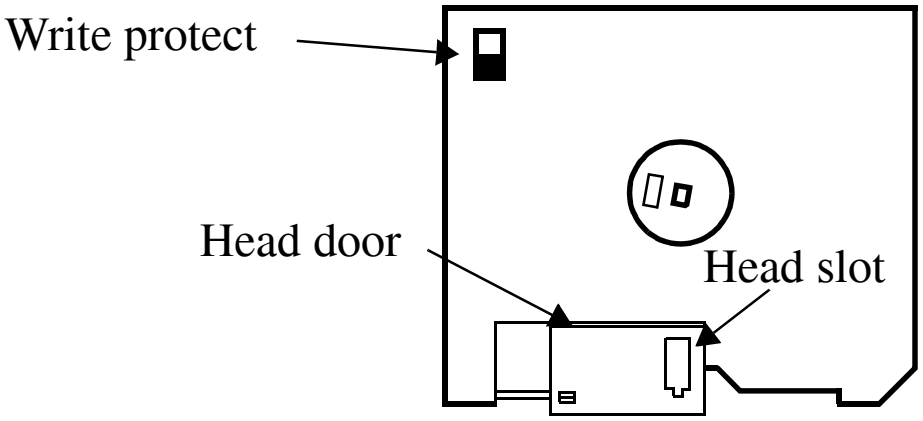
## *Floppy*

The rules are given as follows:

- A data pulse is always stored for a logic 1.
- No data and no clock is stored for the first logic 0 in a string of logic 0s.
- The second and subsequent logic 0s in a row contain a clock pulse, but no data pulse.

The clock is inserted in subsequent 0s to maintain synchronization as data is read from the disk.

The micro-floppy is much more popular today:



## *Floppy*

Advantages of the micro-floppy over the mini-floppy.

- Rigid plastic case provided better protection.
- Head door kept disk from being exposed.
- Write protection mechanism.
- Keyed mechanism for track 0.
- Increase in storage capacity:

  80 tracks *X* 2 sides *X* 18 sectors/track *X* 512 bytes/sector = 1.44 MB.

Extended high density micro-floppy capable of 2.88 MB.

A second extension is the floptical disk which stores data magnetically using an optical tracking system.

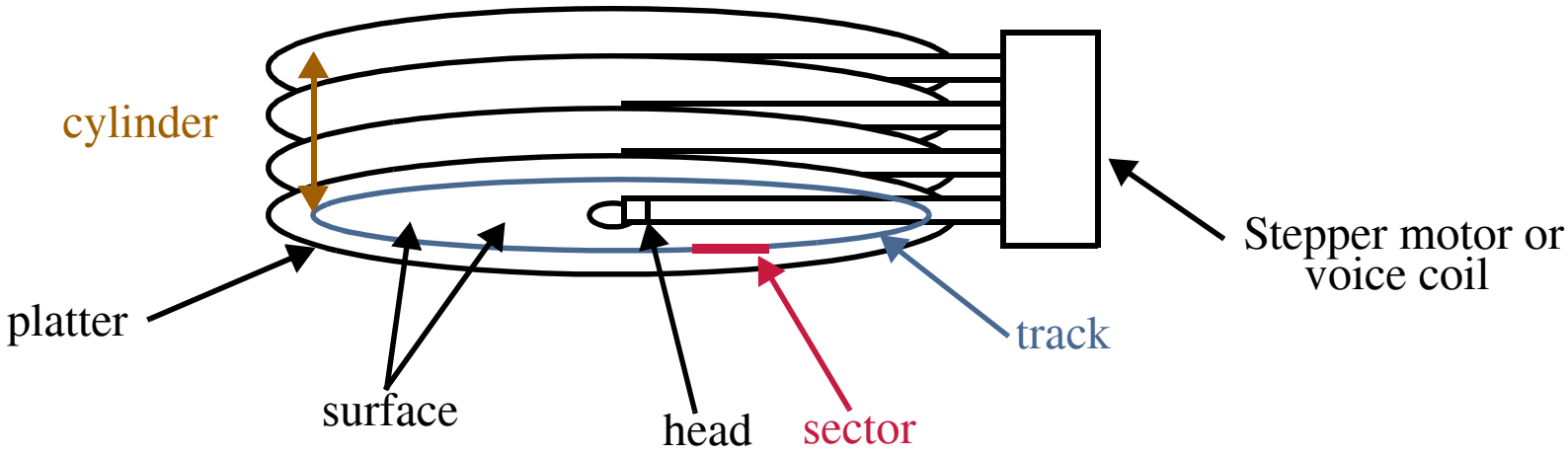
It stores 21 MB of data.

### *Hard Disks:*

Use a flying head to store and read data from the platters and spins at 3,000 to 10,000 RPM (> 10X that of floppies).

## *Hard Disks*

Hard disks usually have at least 4 platters and can have 2 heads per surface.




The heads are moved from cylinder to cylinder using a voice coil.

Hard disks use MFM or RLL (run-length limited) to store information.

RLL 2,7 is common today -- this indicates that the number of zeros in a row is always between 2 and 7.

## *Hard Disks*

The data is first encoded using the table given below.

Note that this encoding always guarantees at least 2 *zeros* and no more than 7 *zeros* in a row.

| Input Data Stream | RLL output |
|---|---|
| 000 | 000100 |
| 10 | 0100 |
| 010 | 100100 |
| 0010 | 00100100 |
| 11 | 1000 |
| 011 | 001000 |
| 0011 | 00001000 |

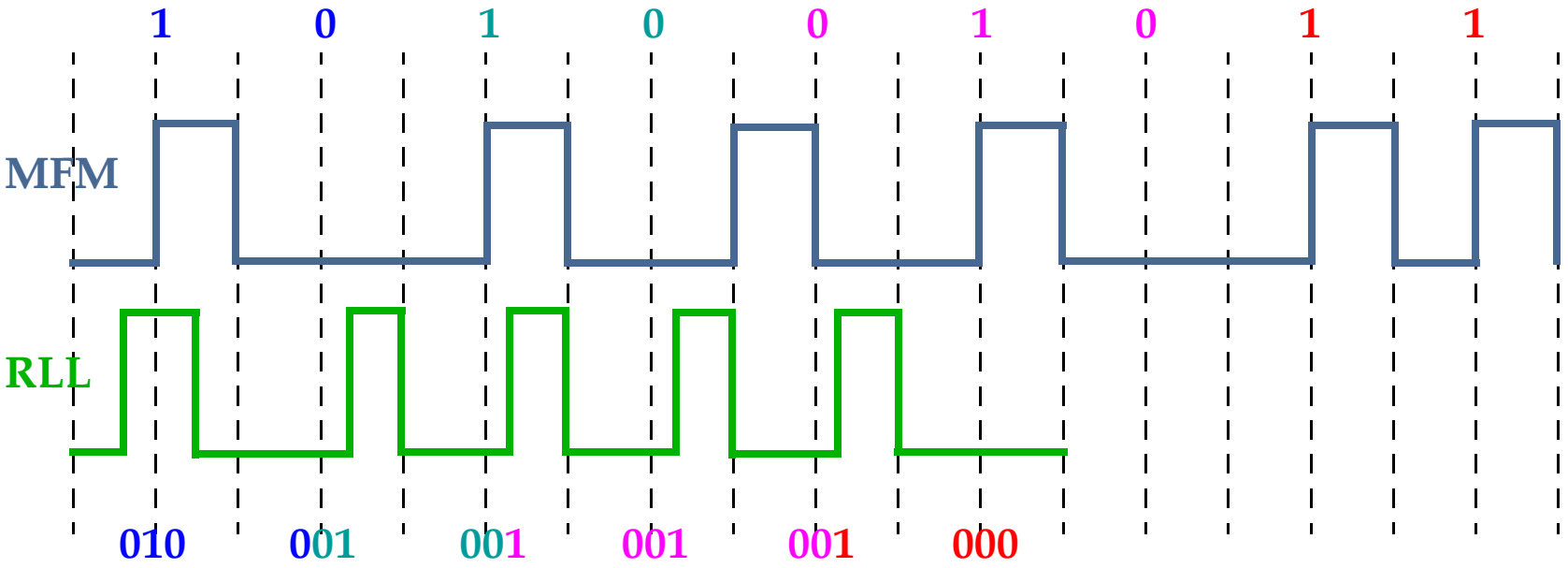
This encoding allows nearly a *50%* increase in storage capacity over MFMs without changing the driver electronics or disk surface.

RLL drives increase the number of tracks from 18 to 27 to achieve this.

40 MB -> 60 MB with better performance.

## *Hard Disks*

For example, given the data stream 101001011:



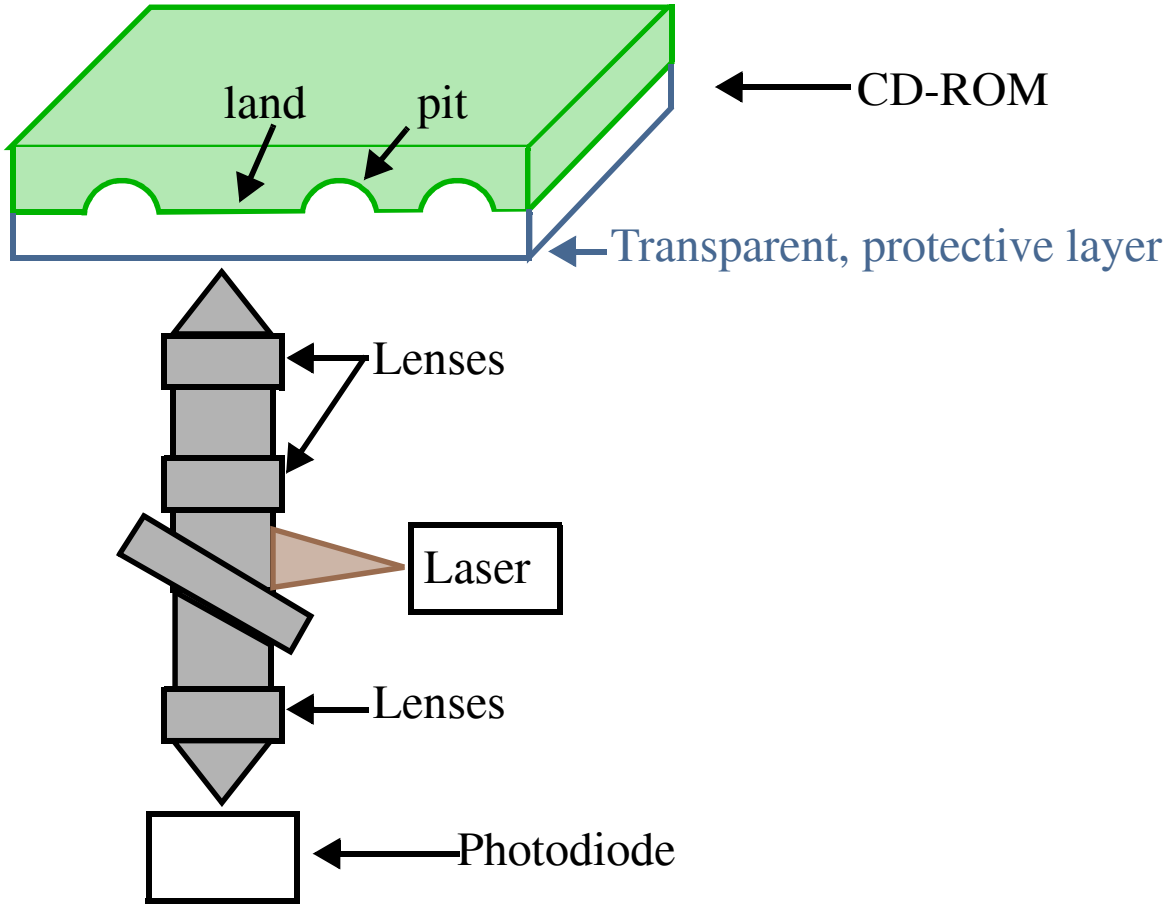

Although all disks use MFM or RLL, disk interfaces vary.

Today's systems use ***ESDI*** (non-existent), ***SCSI*** (small computer system interface) and ***IDE*** (integrated drive electronics).

IDE incorporates the disk controller in the disk drive and usually contain a 32 KB cache.

Access times are less than 10ms (compared with 200ms for floppies).

## *Optical Disks*




CD-ROMs and WORMs store up to 660 MB of data.

DVDs are similar but have much higher bit density (4.7, 8.5 and 17 GB).

## *Video Displays*

Color displays are extremely popular.

Some accept information as a composite video signal (similar to TVs), as TTL voltage level signals (0 or 5V) and as analog signals (0 to 0.7V).

Composites are disappearing since high-resolution cannot be achieved.

They combine the color information with other information such as sync pulses.

Most modern systems use direct video signals with separate sync signals.

Monochrome monitors use one wire for video, one for horizontal sync and one for vertical sync.

Color monitors use three video signals, one for red, green and blue (RGB).

The TTL RGB Monitor:

It uses TTL level signs (0 or 5V) as video inputs and a 4th line called intensity.

It can display a total of 16 different colors (CGA in older systems).

## *TTL RGB Monitor*

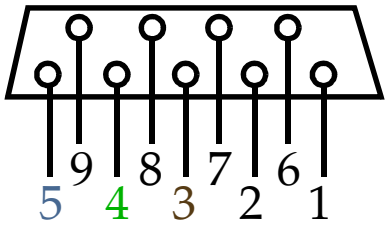
The following table gives the RGB values and colors:

| Intensity | Red | Green | Blue | Color |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Black |
| 0 | 0 | 0 | 1 | Blue |
| 0 | 0 | 1 | 0 | Green |
| 0 | 0 | 1 | 1 | Cyan |
| 0 | 1 | 0 | 0 | Red |
| 0 | 1 | 0 | 1 | Magenta |
| 0 | 1 | 1 | 0 | Brown |
| 0 | 1 | 1 | 1 | White |
| 1 | 0 | 0 | 0 | Gray |
| 1 | 0 | 0 | 1 | Bright Blue |
| 1 | 0 | 1 | 0 | Bright Green |
| 1 | 0 | 1 | 1 | Bright Cyan |
| 1 | 1 | 0 | 0 | Bright Red |
| 1 | 1 | 0 | 1 | Bright Magenta |
| 1 | 1 | 1 | 0 | Yellow |
| 1 | 1 | 1 | 1 | Bright White |

Cyan is a combination of Green and Blue, Magenta - Red and Blue, etc.

## *TTL and Analog RGB Monitor*

The connector pin definitions for either color or monochrome




1 and 2: Ground
3: Red video
4: Green video
5: Blue video
6: Intensity
7: Normal video
8: Horizontal retrace
9: Vertical retrace

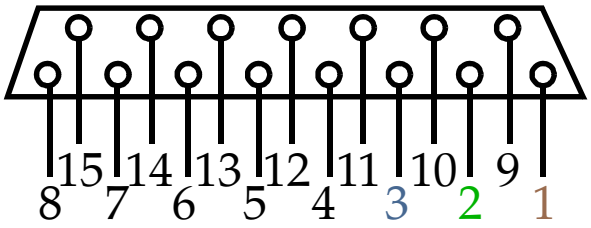
Horizontal and vertical retrace are for synchronization.

Normal video is used for 'intensity' on monochrome monitors.

Analog RGB Monitors

Analog RGB monitors have 3 video signals (no intensity) that can be driven with values between 0 and 0.7 V.

Most can display 256K, 16M or 24M colors.




1: Red
2: Green (mono)
3: Blue
4 and 5: GND
6/7/8: RGB GND
9: Female is blocked
10 and 15: GND
11: Color detect (GND mono)
12: Mono detect (GND color)
13: Horz sync
14: Vert sync
15: GND

## *Analog RGB Monitor*

Most analog displays use a DAC to generate each color video voltage.

A common standard uses a 6-bit DAC for each video signal for 64 distinct voltage levels over 0 to 0.7 V range.

64 X 64 X 64 = 262,144 (256K) colors.

8-bit DACs yield 16M colors.

Conversion time between 25ns and 40ns is required of the DAC.

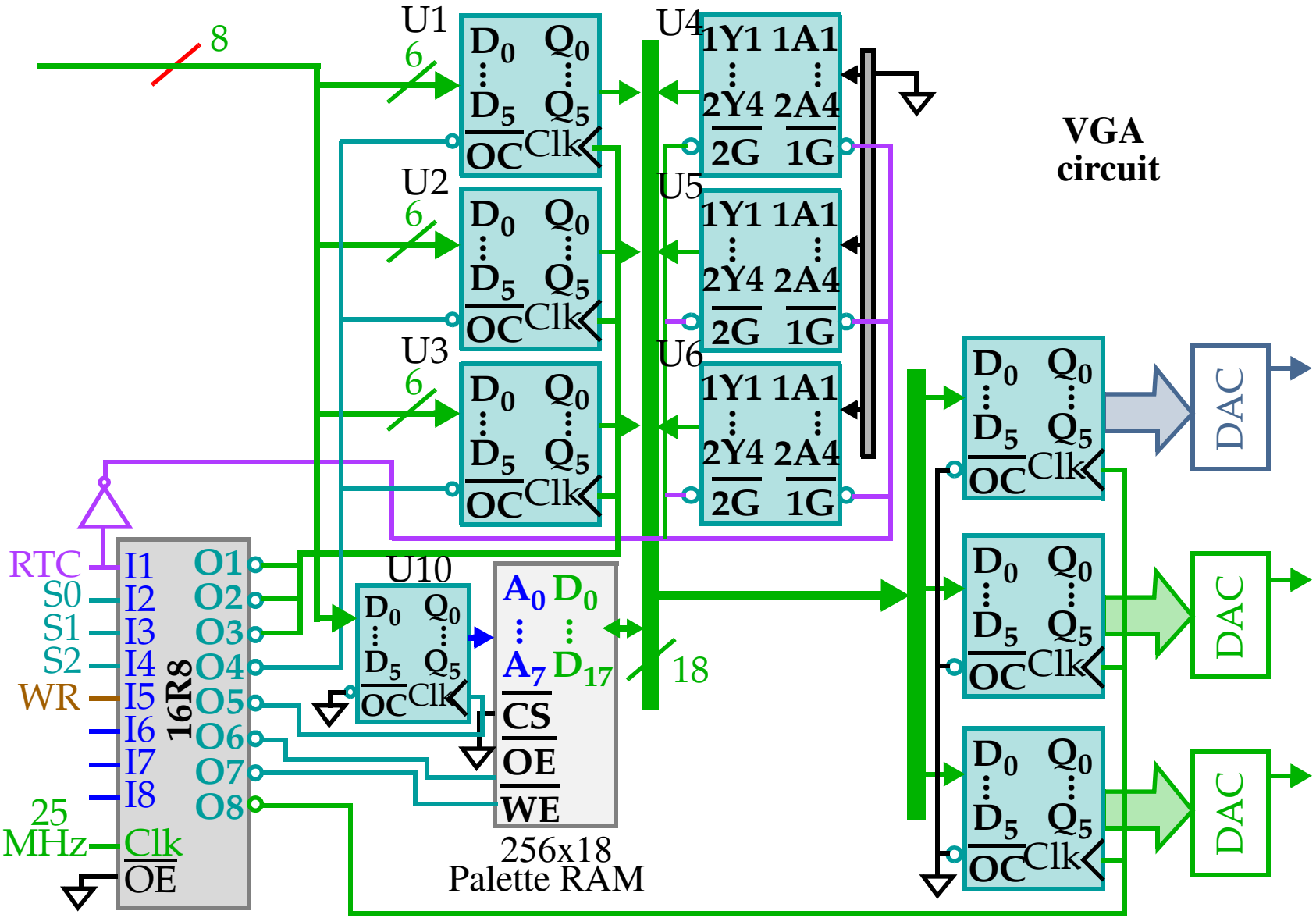
The next slide shows the video generation circuit used in VGA systems.

Each color is generated with a 18-bit digital code (6 each for RG &B).

A high speed palette SRAM (access < 40ns) is used to store 256 different 18-bit color codes (hardware colormap) out of the 256K possible ($2^{18}$).

The 8-bit values (8 bit depth) in the video display RAM specify one of the 256 colors for each pixel position on the screen.

## *Analog RGB Monitor*



## *Analog RGB Monitor*

The 8 bit values from the video RAM are each sent individually on the data bus and latched into U10 by the 16R8.

After 40ns (1/25MHz), the PAL generates a Clk pulse for the DAC latches.
This leaves enough time for the SRAM to access the 18-bit code.

The DACs then convert the 6-bit values to analog voltages for the monitor.

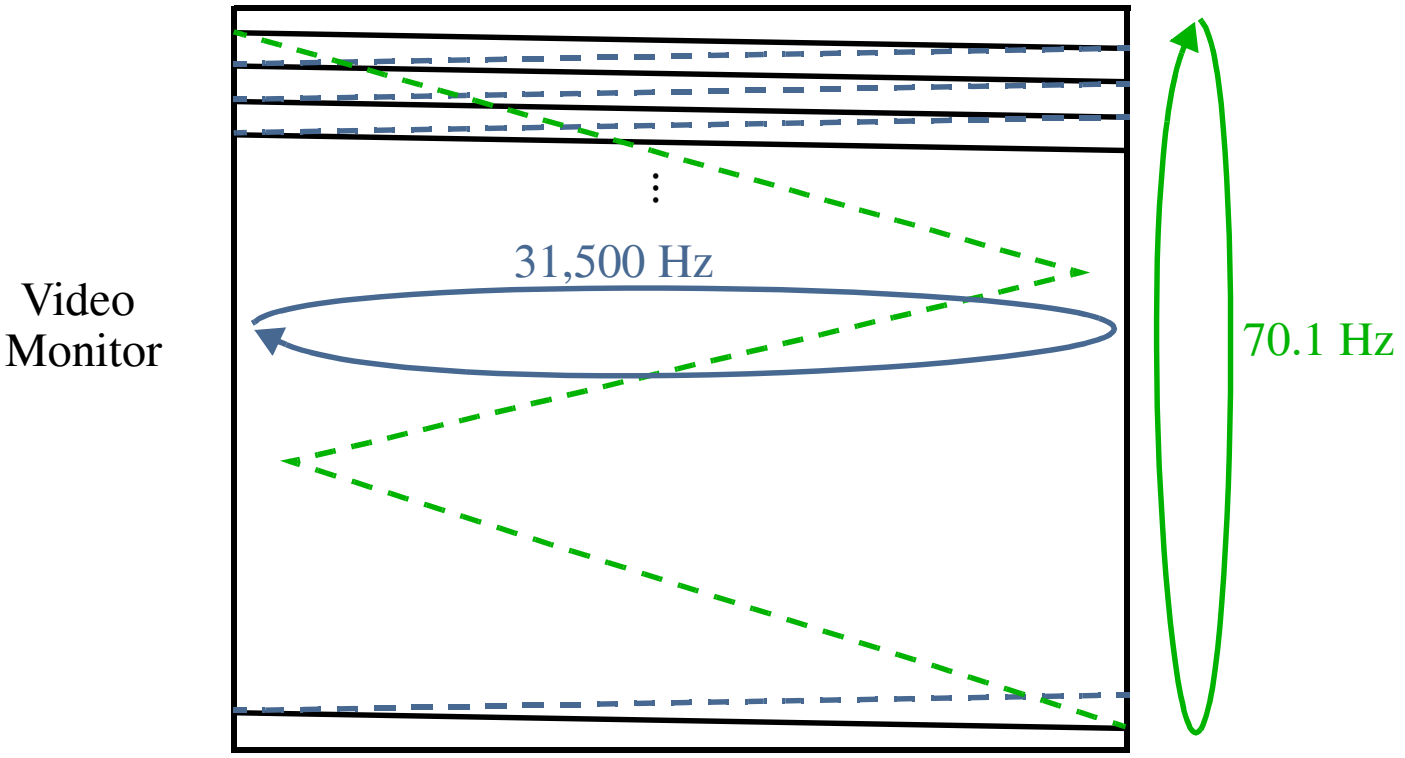
Changing the 18-bit color values is done during retrace, e.g. when RTC is 1.
The PAL latches the address into U10 of the 18-bit cell to overwrite.

Then, S0, S1 and S2 are used to clock each of U1, U2 and U3 in succession as 6-bit values are placed on the data bus.

Finally, $\overline{\text{WE}}$ of the SRAM is pulsed and the U1-U3 outputs are input to the SRAM.

## *Analog RGB Monitor*

Retrace occurs 70.1 times per second in the vertical direction and 31,500 times per second in the horizontal direction for a 640 x 480 display.




During retrace, the video sent to the monitor must be 0 (black).

Buffers U4, U5 and U6 are enabled during this time to force 0.

## *Analog RGB Monitor*

The resolution of the display determine the amount of memory required.

If 256 colors are used (8-bits per pixel) then 640 (width) x 480 or 307,200 bytes of memory are required to store the image.

In order to repaint the 640 pixels of a raster line, 40ns X 640 or 25.6 us are needed.

A horizontal time of 1/31,500 gives 31.746 us.

31.746 - 25.6 = 6.146 us is allowed for horizontal retrace.

Given a vertical retrace of 70.1 Hz, the number of lines repainted is given by (1/70.1)/ 31.746 us = 449.358 lines.

Assume 400 of these lines are used to display information and the rest are lost during retrace.

This leaves 49.358 X 31.746 = 1567 us for vertical retrace.

It is during this time that the palette can be updated or the display RAM is updated with a new image.

## *Direct Memory Access (DMA)*

An alternative to the basic and interrupt-driven I/O discussed previously.

DMA allows data to be transferred between memory and the I/O device without processor intervention.

Speed of transfer limited to speed of memory components or DMA controller (up to 32-40 Mbytes/sec).

Common DMA operations:

- DRAM refresh
- Video refresh
- Disk-memory system reads and writes.

Two signals are used to request/ack a DMA transfer:

- HOLD is an input to the micro that requests a DMA action.
- HLDA is an output from the micro granting the DMA action.

The microprocessor responds by suspending the execution of the program and by placing its address, data and control bus in high-impedance states.

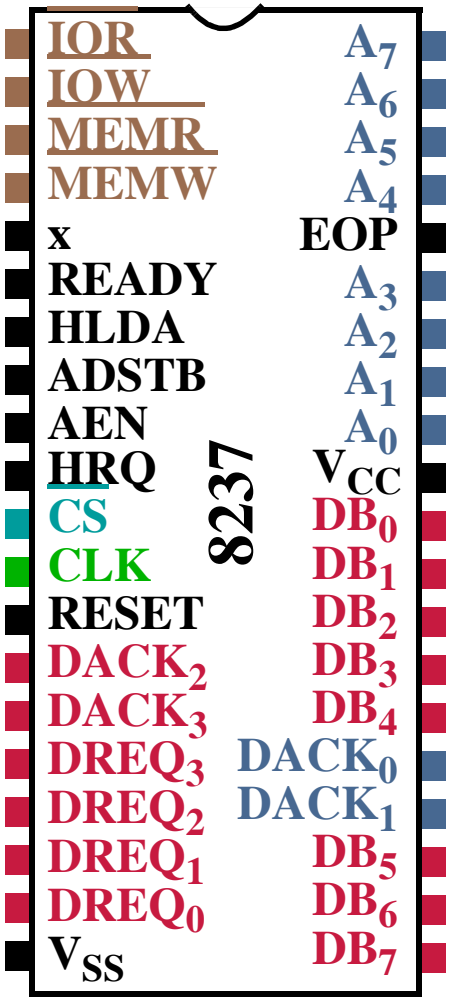
## *Direct Memory Access (DMA)*

DMA 'reads' refer to transfers from memory to an I/O device and involves the use of $\overline{\text{MRDC}}$ and $\overline{\text{IOWC}}$.

DMA 'writes' refer to transfers from an I/O device to memory and involves the use of $\overline{\text{MWTC}}$ and $\overline{\text{IORC}}$.

The data transfer rate is determined by the speed of memory or the DMA controller (usually the latter).

The DMA controller provides memory with the address and select the appropriate I/O device (via $\overline{\text{DACK}}$).



## *DMA Controller: 8237*

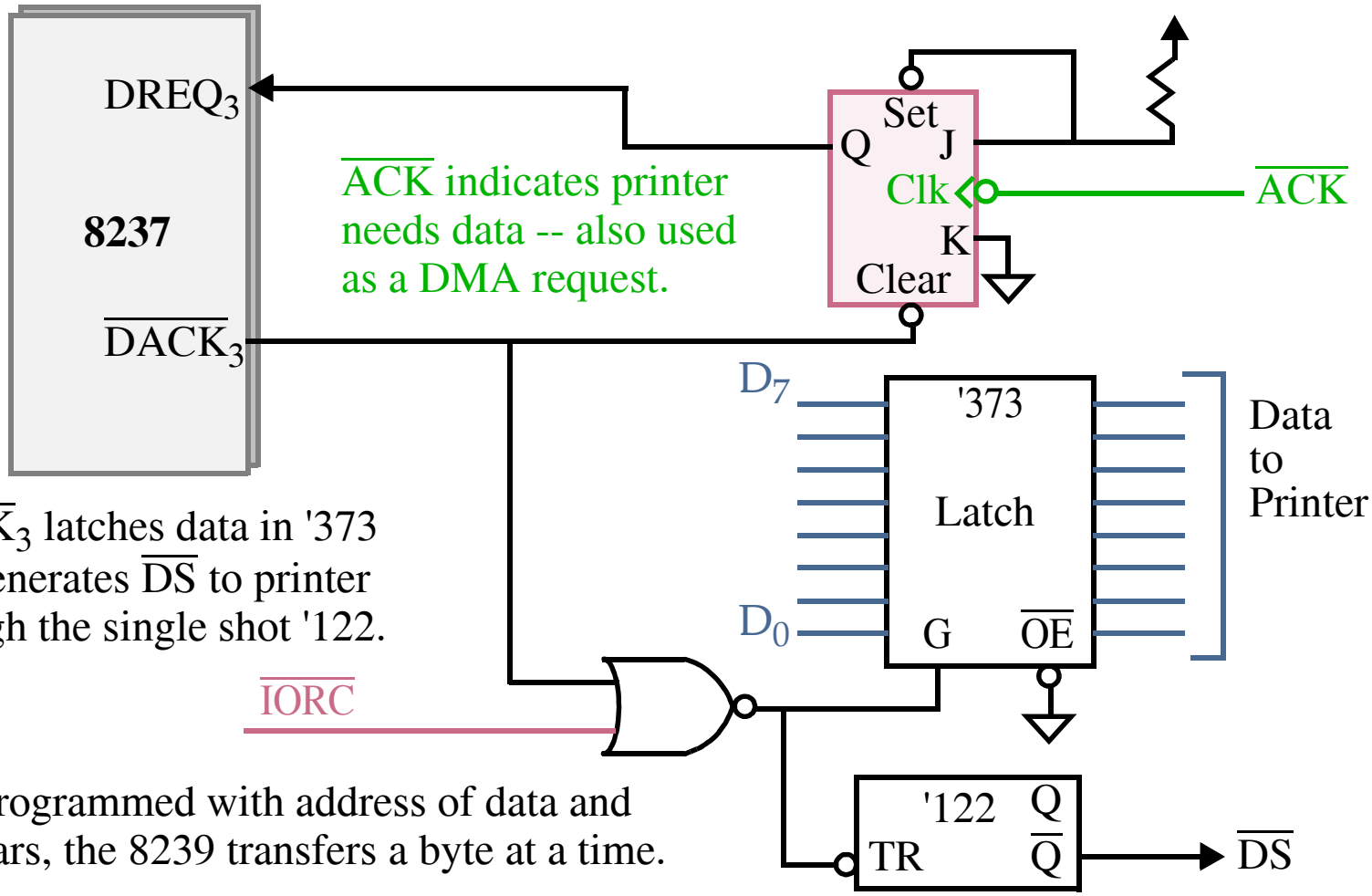
- Clk: < 5MHz.
- $\overline{CS}$: Output of a decoder.
- RESET: Clears all internal registers (command, status, request, etc.).
- READY: Allows memory and I/O to insert wait states into the 8237.
- HLDA: Input that tells 8237 that micro has released address, data and control buses.
- $DREQ_3$-$DREQ_0$: DMA request inputs used to request a DMA transfer.
- $DB_7$-$DB_0$: Used to program the 8237 and output upper 8-bits of address.
- $\overline{IOR}$, $\overline{IOW}$, $\overline{MEMR}$, $\overline{MEMW}$: Outputs used to control memory and I/O.
- $\overline{EOP}$: Bidirectional: as an input, used to terminate a DMA transfer, as an output, signals the end of the DMA transfer.
- $A_3$-$A_0$: Address pins select an internal register during programming and output part of the address for a transfer.
- $A_7$-$A_4$: Address outputs.
- HRQ: Output that connects to HOLD pin on micro to request a DMA.
- $DACK_3$ - $DACK_0$: Used to select an I/O device (ack a DMA request).
- AEN/ADSTB: Enable latch (and strobe) to transfer $DB_x$ to upper 8 A bits.

## *DMA Controller: 8237*

Some of the internal registers are:

- $CAR_3$ - $CAR_0$: Used to hold the 16-bit memory address used for a DMA transfer. Either incremented or decremented after a byte is transferred.
- $CWCR_3$ - $CWCR_0$: Current word count register programs a channel for the # of bytes (up to 64KB) transferred during a DMA action.
- CR: Command register programs the operation of the 8237. Bits in this register allow:
  - Memory-to-memory transfers (like MOVSB) where DMA channel 0 holds the source address and DMA channel 1 holds the dest address.
  - Memory-to-memory transfers in which DMA channel 0 holds a constant address -- used to fill a memory regions with a constant.
  - Fixed or rotating DMA channel priority, plus misc other options.
- MR: 'Mode of operation' register -- one for each channel. For example, block mode is used for memory-to-memory transfers.
- RR: Request register is used to request a DMA transfer via software -- essential for processor initiated memory-to-memory transfers.
- SR: Status register indicates when a DMA has completed.

## *DMA-Processed Printer Interface*




$\overline{DACK_3}$ latches data in '373 and generates $\overline{DS}$ to printer through the single shot '122.

Once programmed with address of data and # of chars, the 8239 transfers a byte at a time.

Note that the I/O device is NOT selected by decoding the address bus, but rather by $\overline{DACK}$, since address bus contains a memory address.

See code in book and example of 8237 connected to an 8088.