



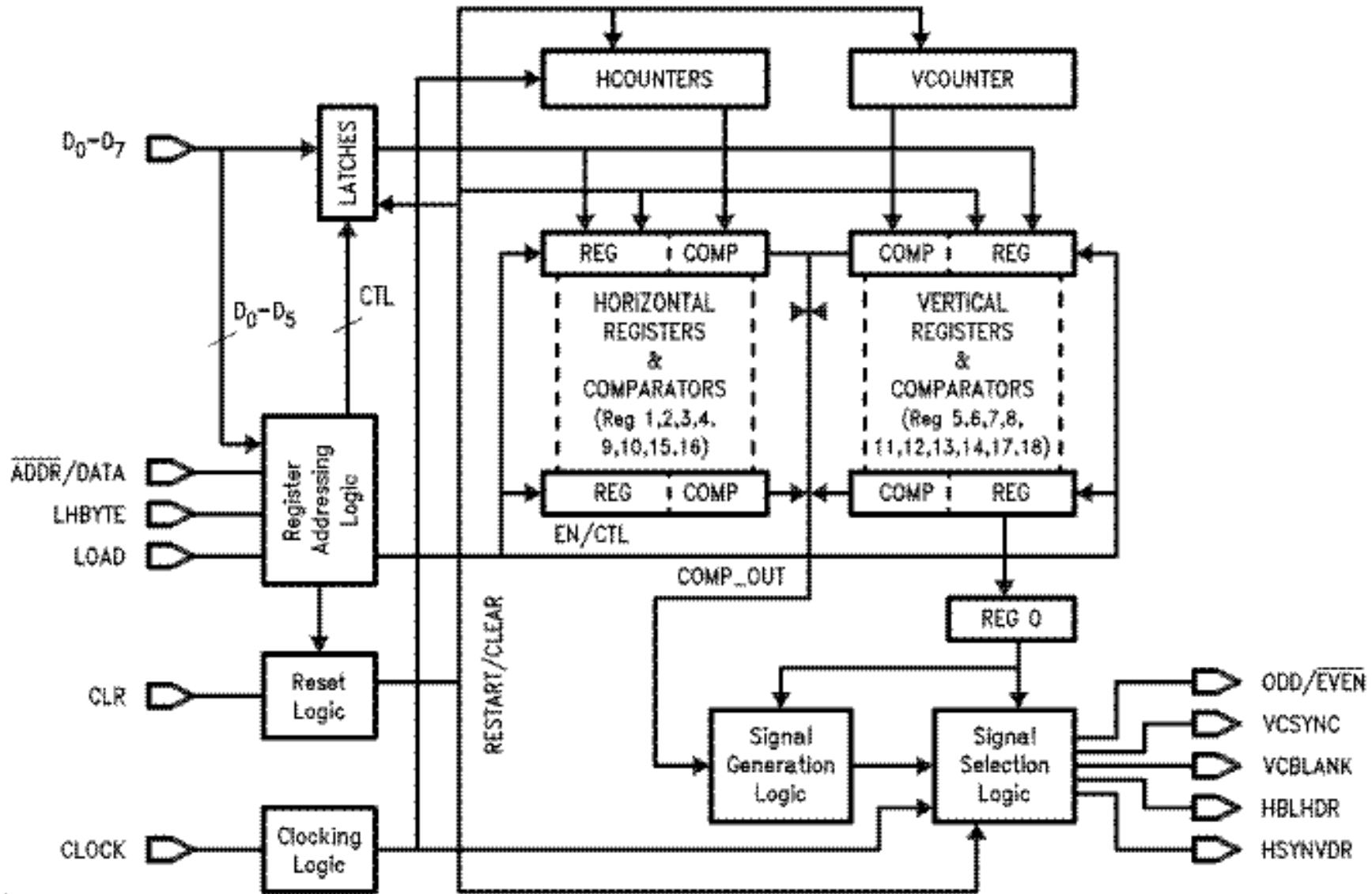
Video, Quiz Review, and Project Clues



LM 1882 Sync Generator



See web page. – Look under Resources.– Click on Video sync.

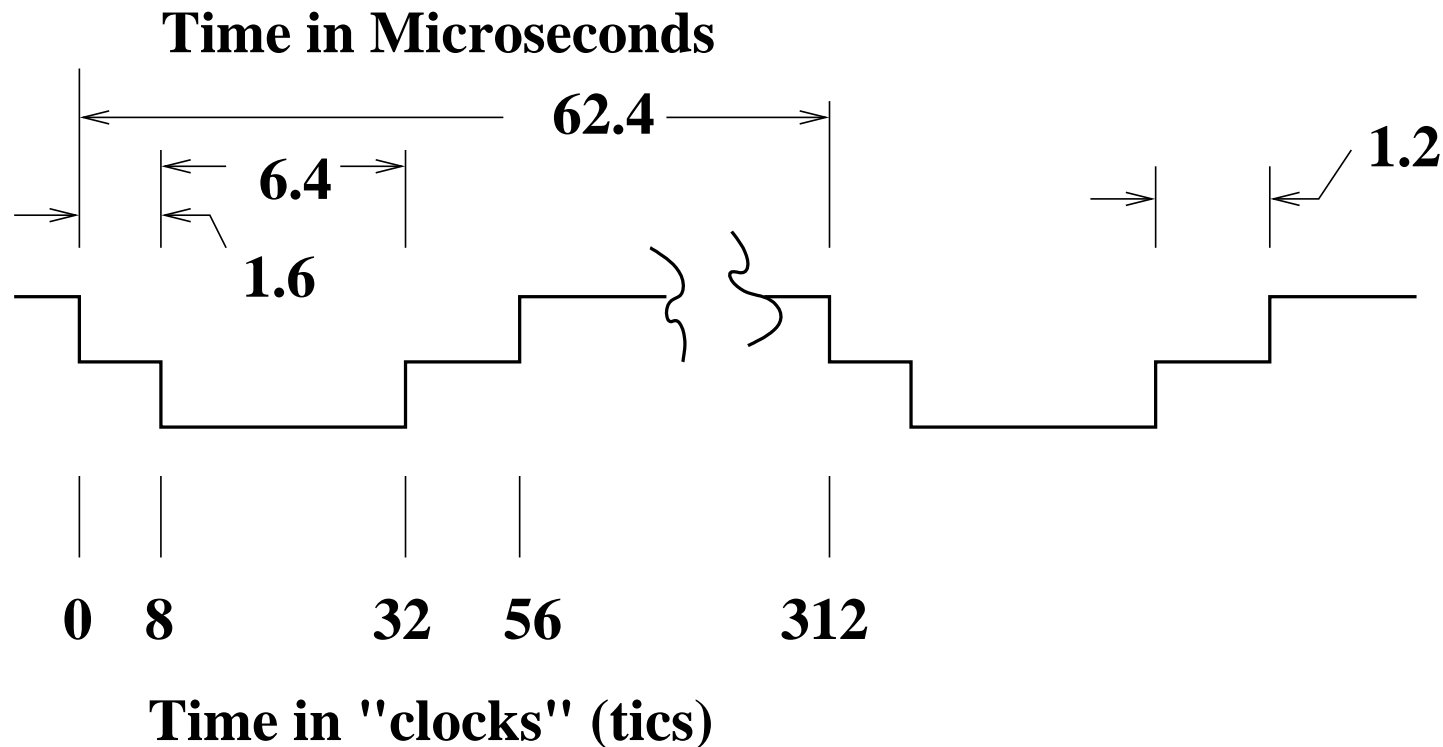




Example Timing



- LM1882 is flexible: timing information is stored in registers.
- See data sheet no. 95 for particulars.
- One example is:
 - 256 pixels wide
 - 256 lines
 - 5 MHz clock (probably not typical)





Sync Generator: Register Contents



Register Contents:

Horizontal (Line) Control

| | | |
|-----------|------------|----------------------------------|
| R1 | 9 | Horizontal Front Porch |
| R2 | 33 | Horizontal Sync Pulse End |
| R3 | 57 | Horizontal Blanking |
| R4 | 312 | Line Width – must be even |

} Time in "clocks"

Vertical (Frame) Control

| | | |
|-----------|------------|---|
| R5 | 4 | Vertical Front Porch |
| R6 | 7 | Vertical Sync Pulse End |
| R7 | 21 | Vertical Blanking |
| R8 | 276 | Frame: 256 lines + 20 lines blanking |

} Lines

Register 0: Contents 0110 0001 1000

| | |
|--|-------------|
| Bit 10: Enable System Clock | 1 |
| Bit 9: Disable Equalization | 1 |
| Bits 8:5 Sync Pulses Active Low | 0000 |
| Bits 4:3 Non-Interlaced | 11 |
| Bits 2:0 Default Output Config: | 000 |

Pin 12 CBLANK, Pin 13 HGATE, Pin14 CSYNC, Pin 15 VGATE

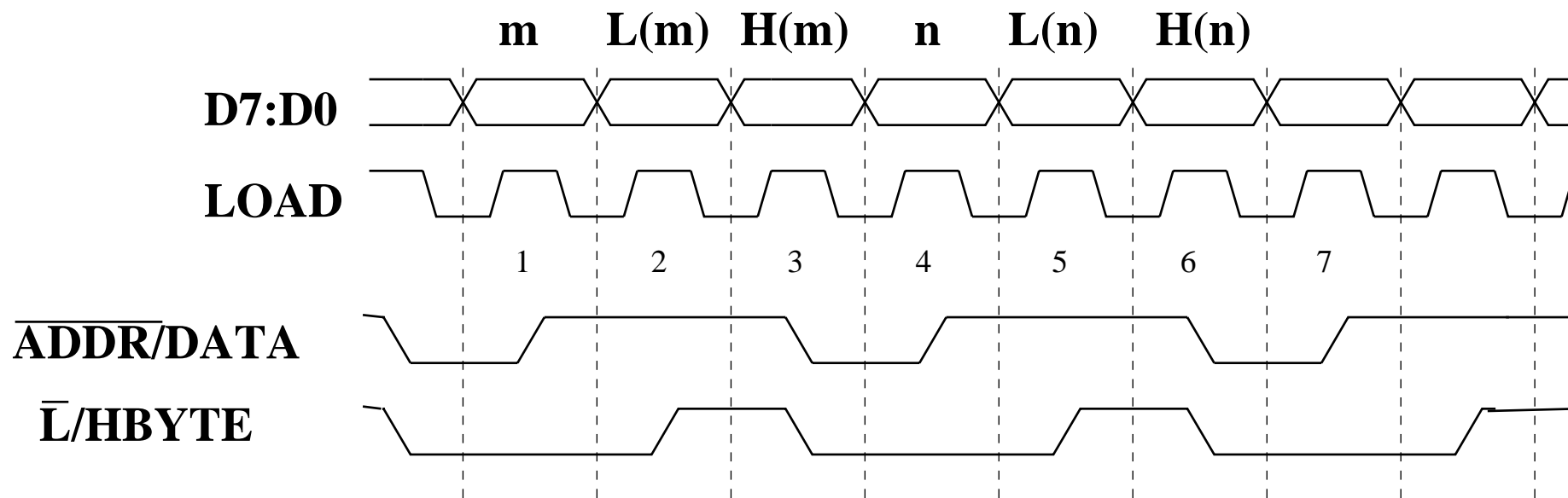


Sync Generator – Manual Mode



- LM1882 must be loaded on power up.

| Cycle No. | Load Falling Edge | Load rising edge |
|-----------|--------------------------|------------------|
| 1 | Enable Manual Addressing | Load address m |
| 2 | Enable Lbyte Data Mode | Load Lbyte m |
| 3 | Enable Hbyte Data Mode | Load Hbyte m |
| 4 | Enable Manual Addressing | Load address n |
| 5 | Enable Lbyte Data Mode | Load Lbyte n |
| 6 | Enable Hbyte Data Mode | Load Hbyte n |



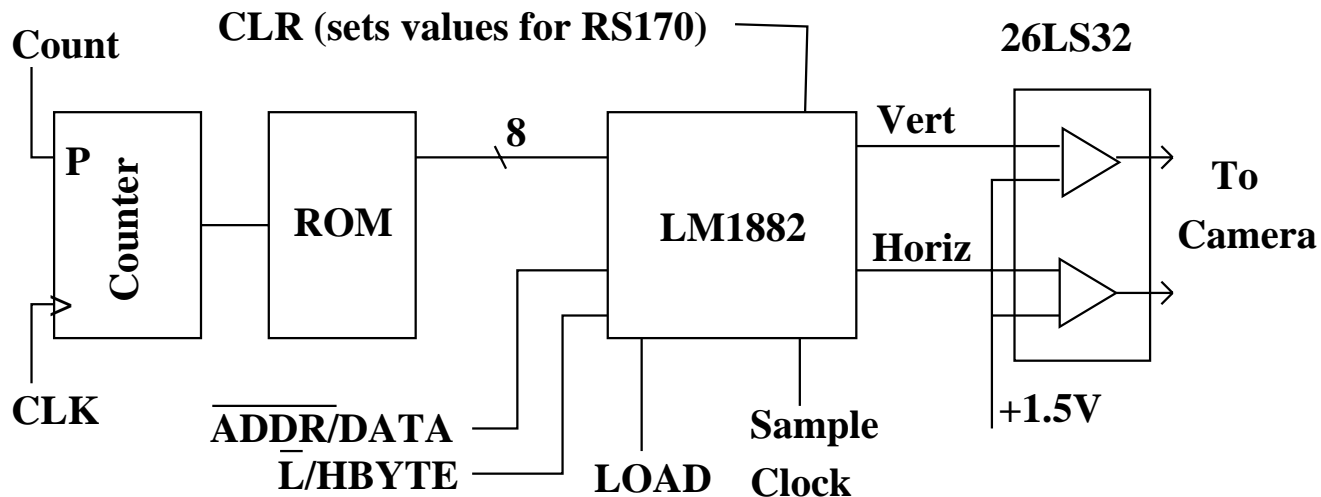


Sync Generator – Automatic Mode



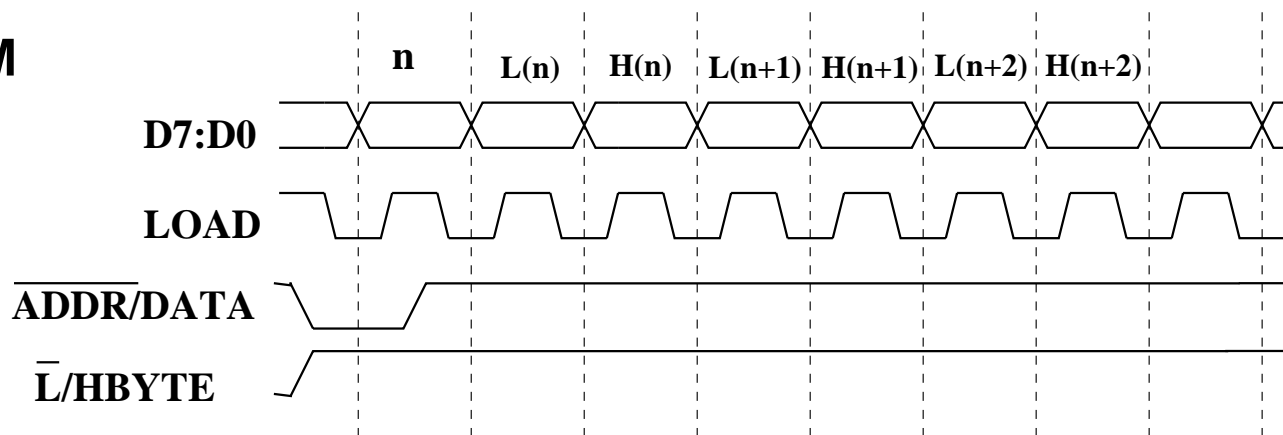
LM 1882 must be loaded on power up.

Use a ROM \ (or PROM) to hold configuration values.



Instantiate the ROM from LPM (FPGA).

Design a (minor) FSM to do the programming of the LM1882 registers.





Do Not Use the LM 1882



- **The LM 1882 is a general purpose sync generator designed to be run by a microprocessor.**
 - You likely want to use a sync generator in one mode only.
 - You do not need all of the registers.
 - They provide generality you likely do not need.
 - Avoid problems associated with loading the registers on power up.

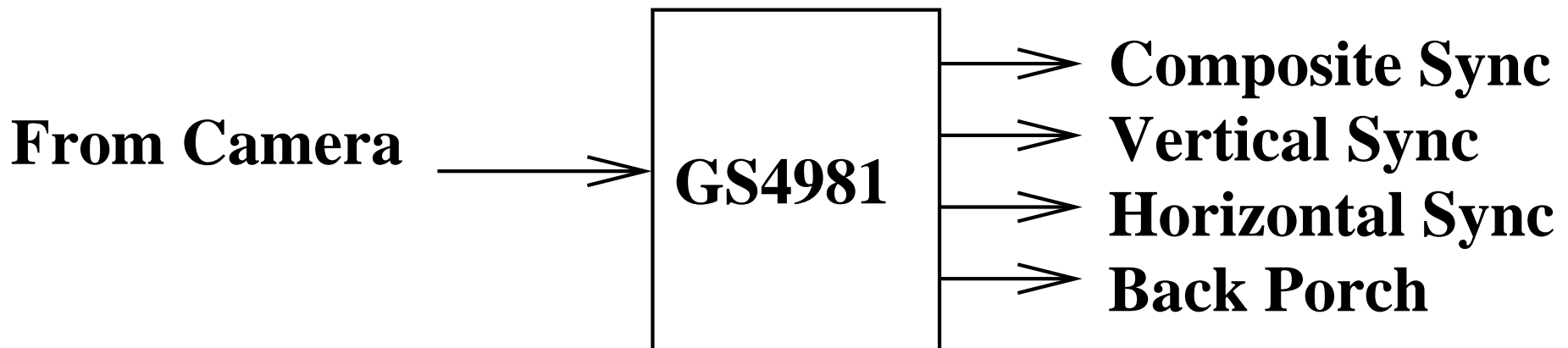
- **Use the organization and block diagram of the LM 1882 as a guide to designing your specific sync generator.**
 - Encode specific x and y address counters in VHDL.
 - This way you can generate exactly the sync signals that you want.



Sync Separator



- A sync separator operates in the reverse direction.
 - GS4981 generates composite sync from video.
 - It also generates separated sync signals.
- The sync separator is not easy to implement in VHDL as its input is an analog signal.
- However, your pixel clock must be synchronized with the recovered horizontal sync.
 - If you do this synchronization with the pixel clock signal directly, then the pixel clock used will “crawl” a whole pixel time.
 - It is better to use a faster clock, say 4 times, to do the synchronization and then the “crawl” will only be $\frac{1}{4}$ of a pixel time (distance).

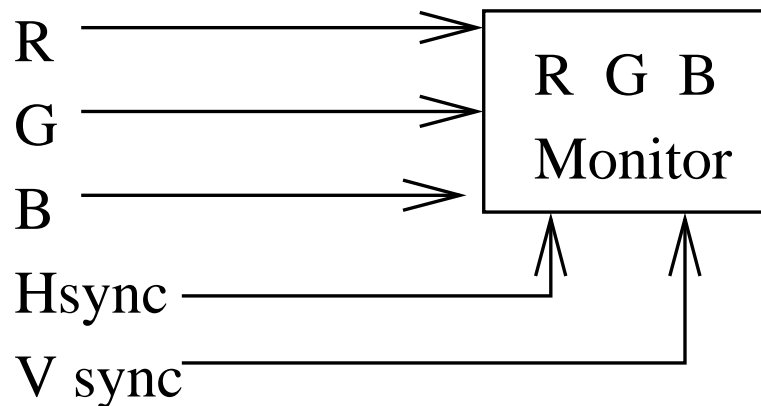




Color Displays



- Color displays are similar to three monochrome displays operated together, i.e., the colors add.
- Three binary signals yield an eight color display.
 - Some monitors have an analog video input for each color.
- Sync is sometimes on a separate wire.
 - Sometimes it is superimposed on the green signal.



| | R | G | B |
|--------|---|---|---|
| Black | 0 | 0 | 0 |
| Blue | 0 | 0 | 1 |
| Green | 0 | 1 | 0 |
| Cyan | 0 | 1 | 1 |
| Red | 1 | 0 | 0 |
| Purple | 1 | 0 | 1 |
| Yellow | 1 | 1 | 0 |
| White | 1 | 1 | 1 |

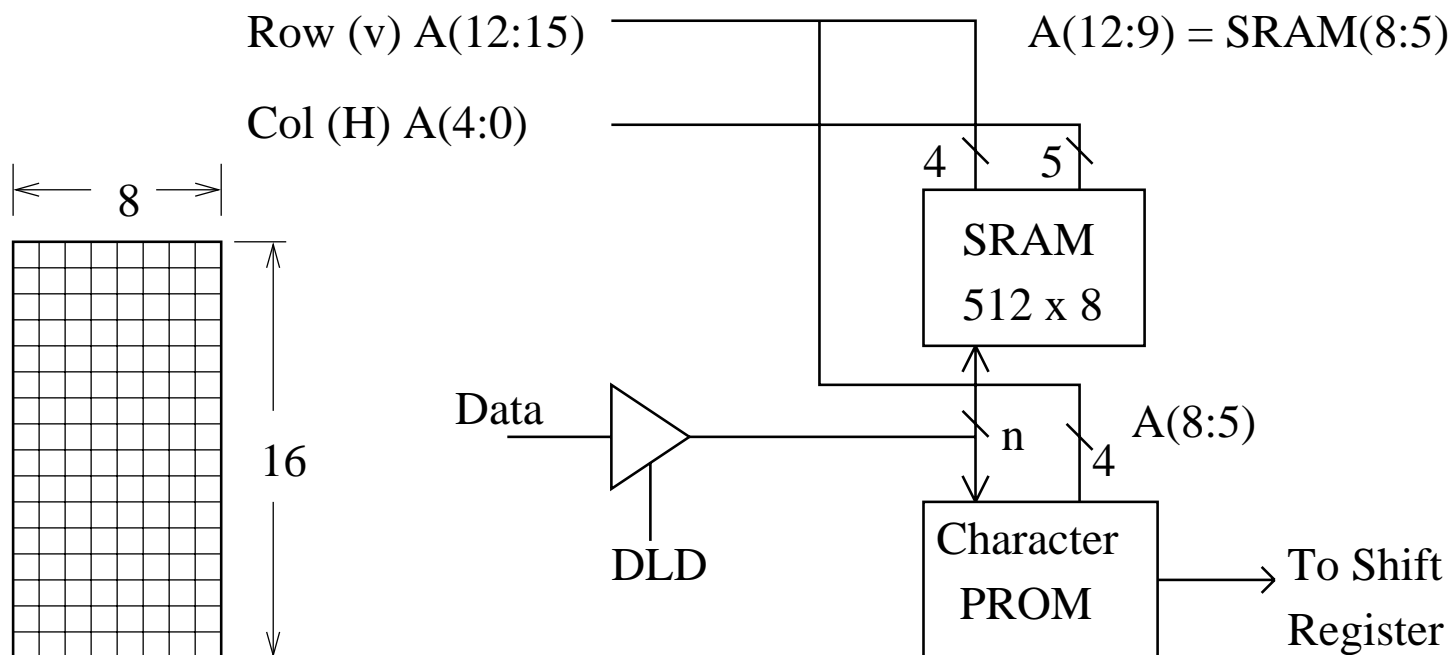


Character Display (8 x 16 pixels)



- Characters are fixed bit patterns.
 - They always have the same shape but can appear at different places on the screen.
 - Use of characters can save video memory and make the manipulation of video memory contents simpler.

For a screen 256 x 192 one gets 384 characters. The screen address is used to specify the position and part of the address of the character ROM

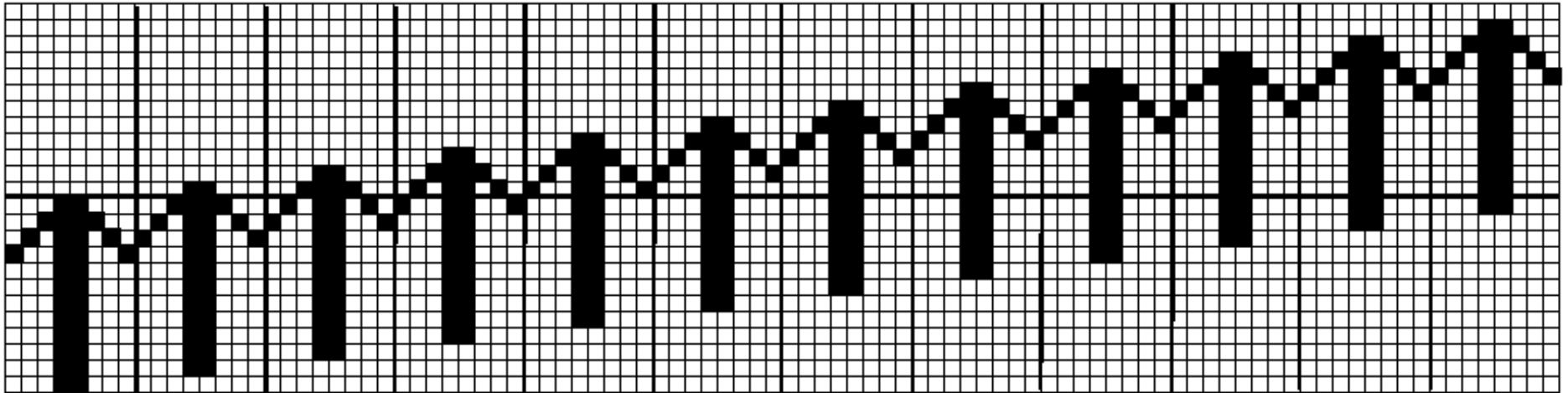




Pairs of Characters



- Sometimes, pairs of characters can create the same motion effect as bit mapped graphics.
 - The speed of the motion depends on the update rate.
- These 24 characters (12 x 2) can display an arrow at any vertical position.

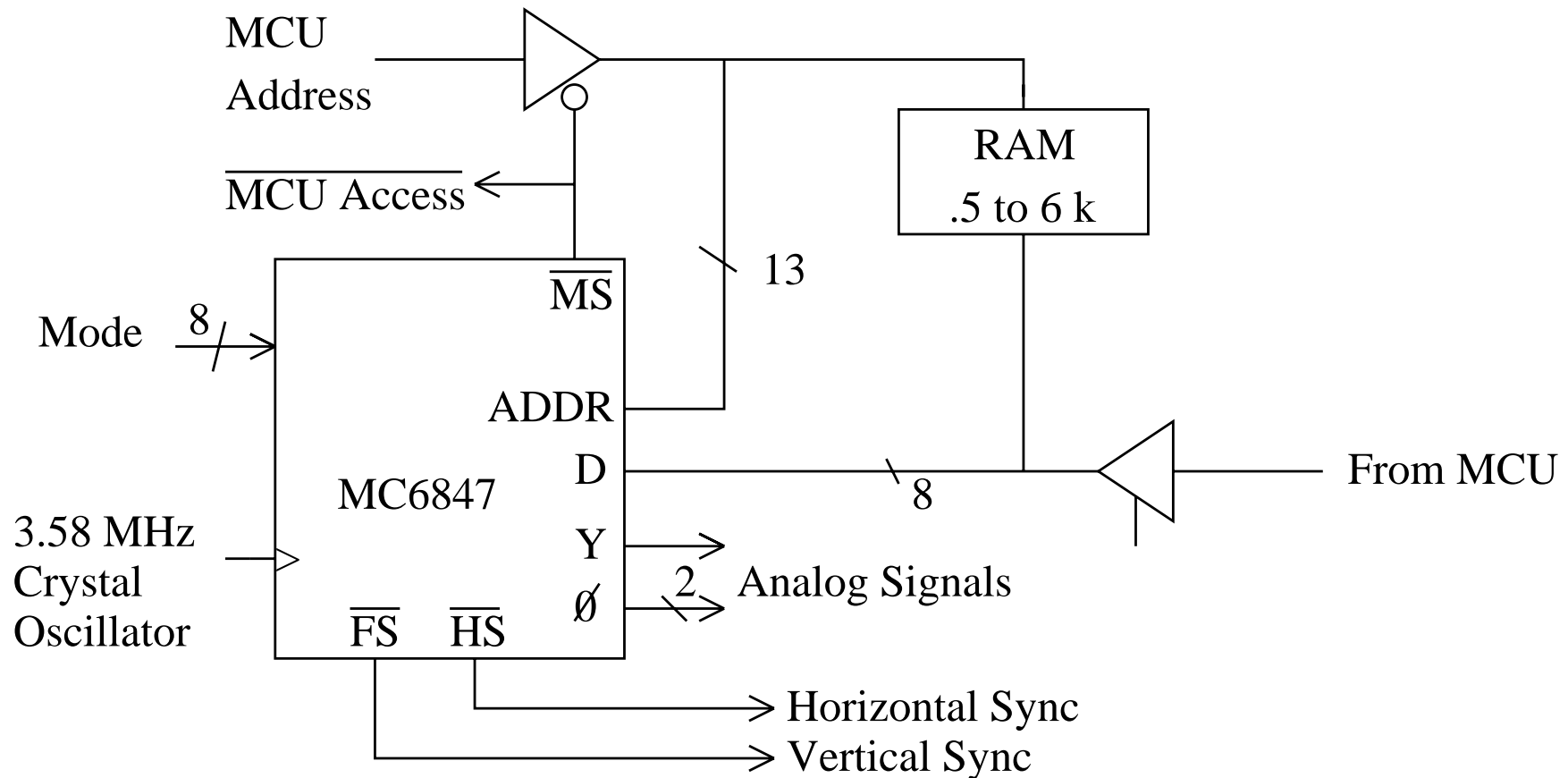




Video Controllers



- **MC6847 is obsolete (but available).**
 - It provides a 13 bit address and an analog video signal.
 - It reads 8 bit data which can be either a character code or video data.
 - Several display modes include 256 x 192 two color and several other color graphics modes with lower resolution.

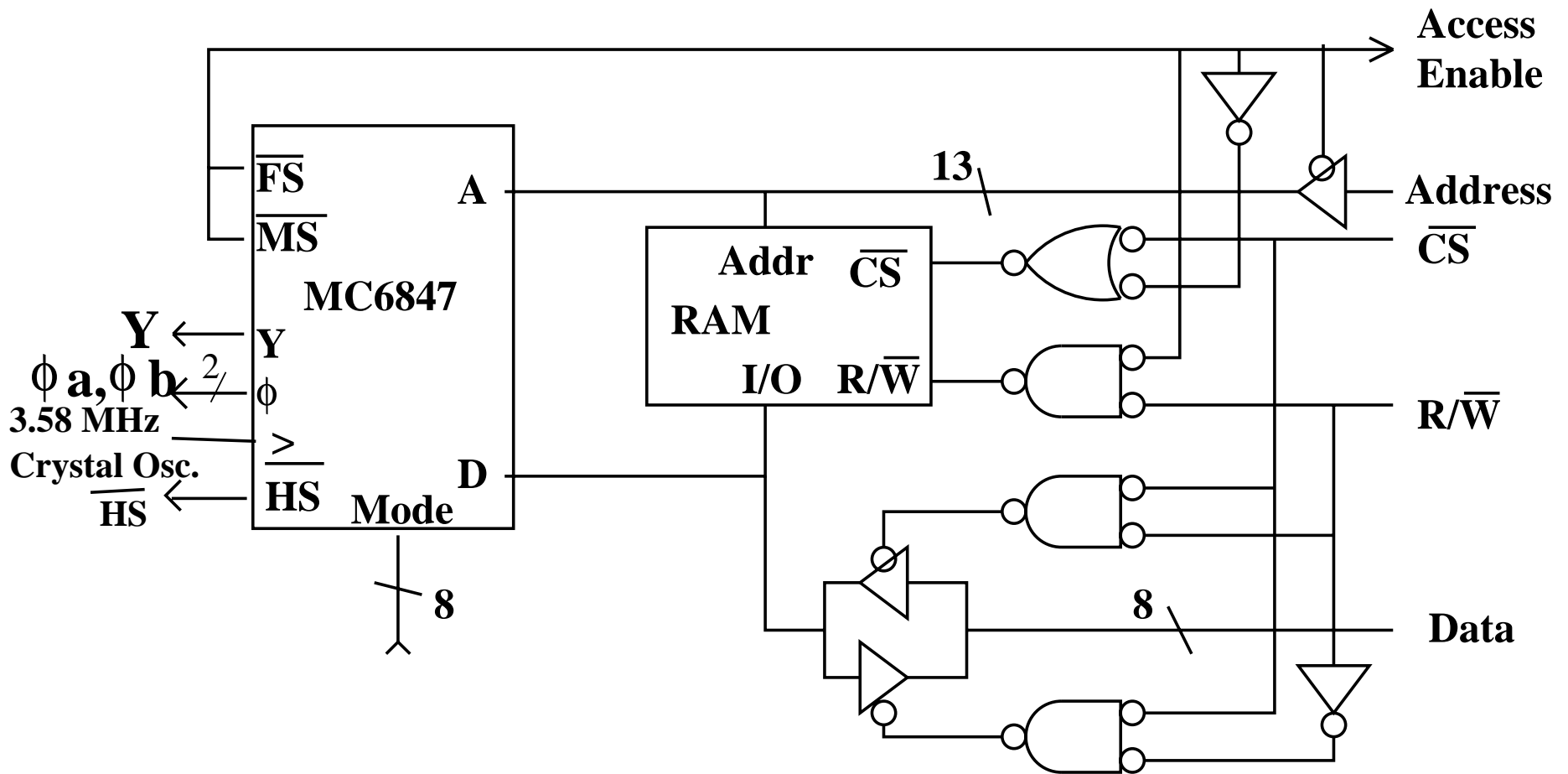




Bit Mapped Video



- MC6847 can be used with bit mapped video.
 - More about Y, phi A, and phi B later.



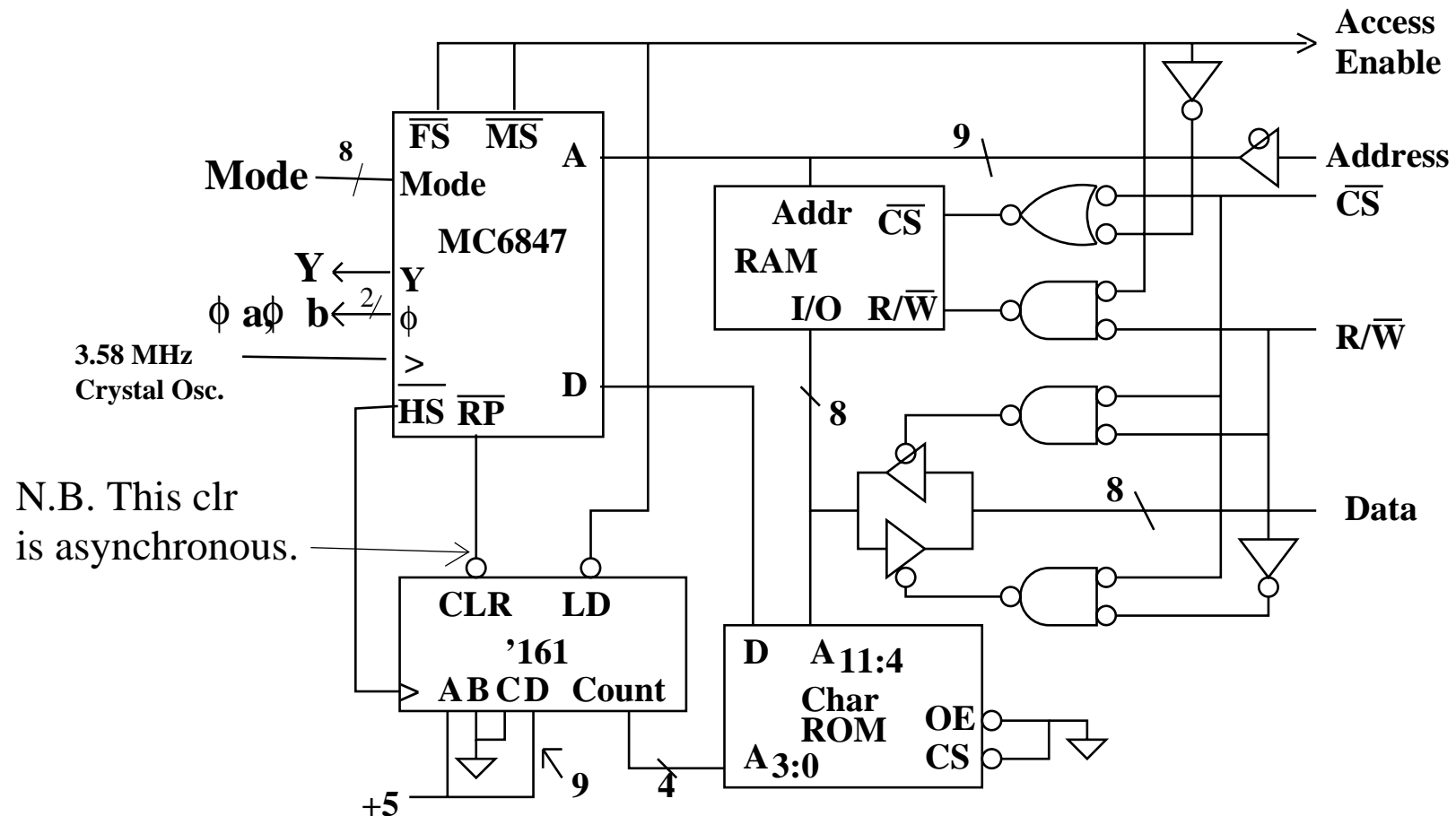


MC6847 With a Character ROM



■ You can call it a bug or a feature!

- To get around the fact that \overline{RP} begins and ends between \overline{HS} pulses, one must use a counter that has an asynchronous clear, similar to the '161. Of course, one can implement this counter with VHDL but one has to remember to implement an asynchronous clear.

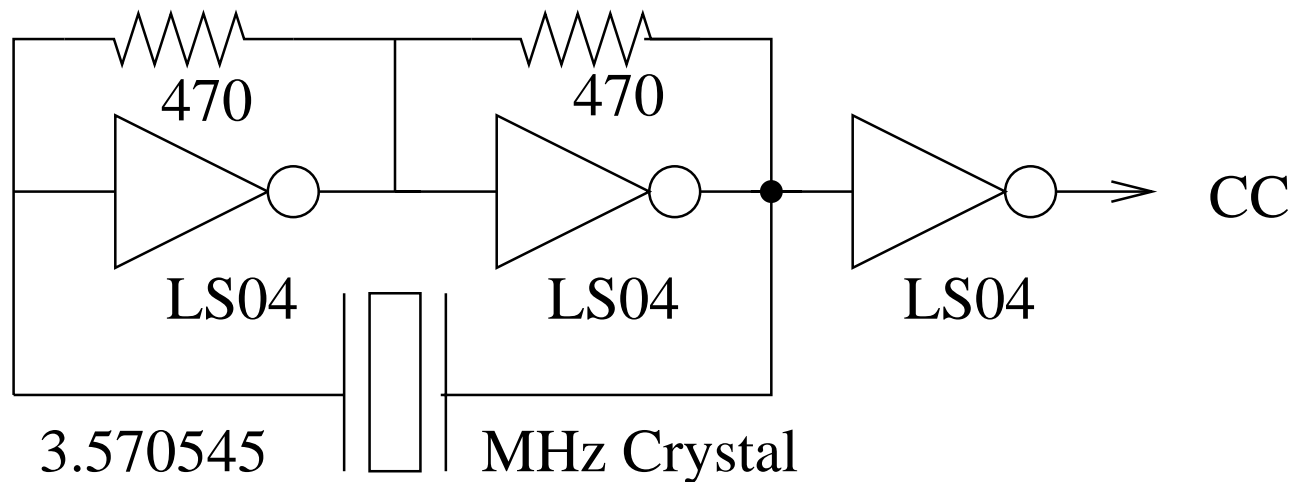




Supplying a Clock



- The MC6847 is meant to be run by a color burst crystal oscillator.
- The recommended circuit is to use a pair of LS04 inverters. to provide the gain for an oscillator.

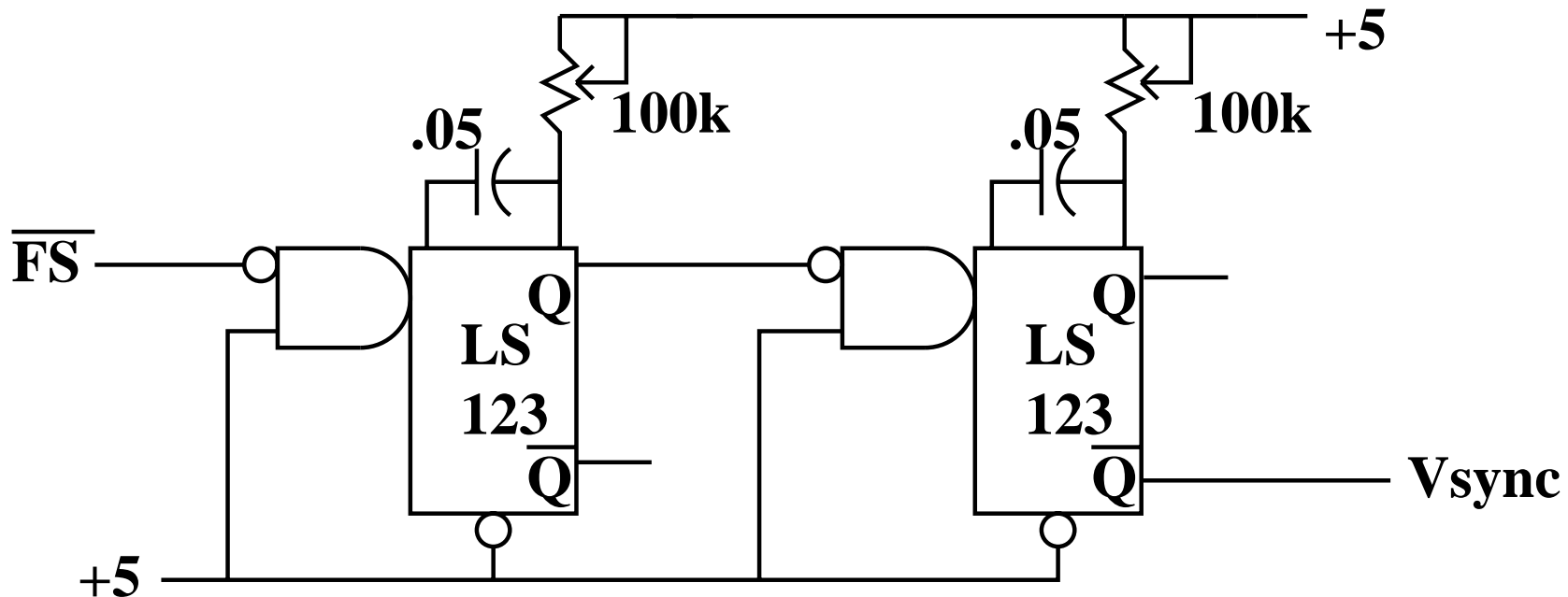




Centering the Display



- The MC6847 generates blanking, not vertical sync.
- You have to generate vertical sync.
 - You can do this digitally with counters.
 - Alternatively, one can use two one shots.
 - The first one shot triggers on the falling edge of /FS and determines the delay of Vsync.
 - The second one shot determines the width of Vsync.
 - Adjust the pots so that the picture is centered on the monitor.

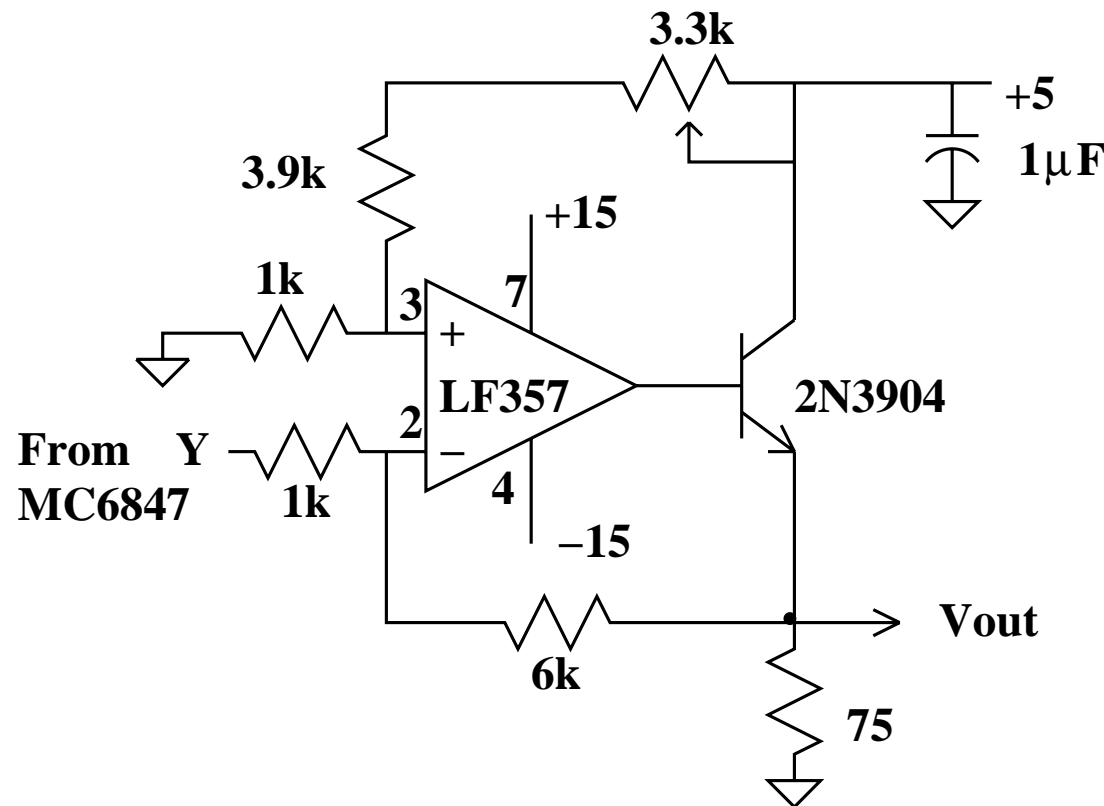




Drive a Monitor



- Use a 75 ohm coax cable to connect to the monitor.
 - Remember to terminate the monitor with 75 ohms.
 - There is usually a switch on the back of the monitor.
- Here is a simple way to drive a monochrome monitor.
 - Don't make the gain of the '357 too small or you will get some ringing.
 - The Y output has composite sync and luminance.

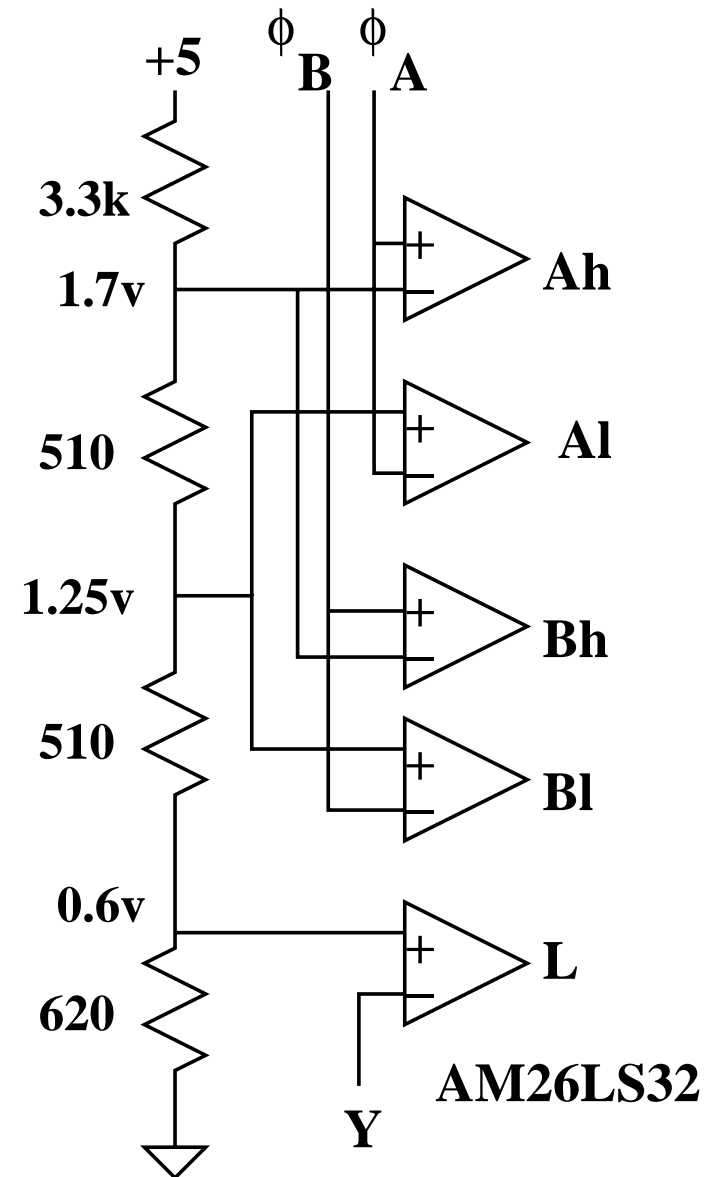




Color Output



- The first step is to convert the analog signals, phi A and phi B, to digital signals.
- The circuit on the right works for one of the color modes. It is not guaranteed to work for the mode you want to use!
 - In particular, the luminance signal may never be a one for your mode.
 - Display a test pattern and look at the analog signals with a scope.
- The resistor chain determines the voltages input to the comparators (as in a flash converter).
- The AM26LS32 is a “fast” comparator (it is actually a line receiver).





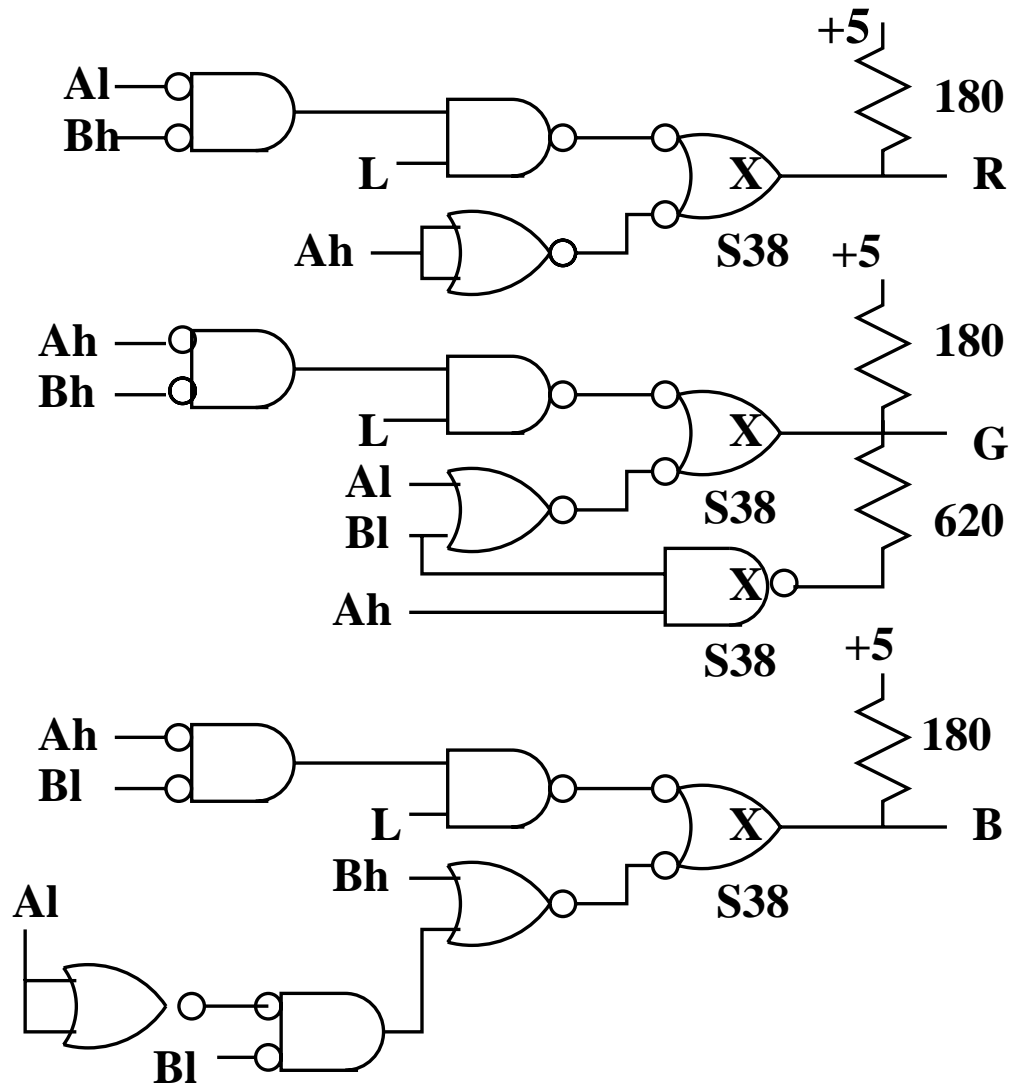
Coding to Get RGB



- This combinational logic can (should) be implemented in VHDL.
- The logic depends on the mode selected.
 - Look at the Ah, Al, Bh, Bl, and L outputs to determine the logic needed.
 - The 74S38 driving the 620 ohm resistor is one way to get orange.

An available cable pinout is:

| | |
|-------|-----------|
| Pin 1 | Intensity |
| Pin 2 | Red |
| Pin 3 | Green |
| Pin 4 | Blue |
| Pin 5 | GND |
| Pin 6 | GND |
| Pin 7 | HSYNC |
| Pin 8 | VSYNC |



These Drive RGB Inputs



Code Your Own Controller



- **It is better to code your own controller in VHDL.**
 - The 6847 requires some extra hacks.
 - Clock generation
 - Centering
 - “Analog to Digital” conversion of chrominance and luminance signals
 - Combinational logic to convert to RGB
 - Asynchronous peculiarity of row counter

- **Use it as a guide to form a block diagram of what you want.**
 - Pick the mode or modes that you want to use.
 - Determine RGB signals directly by decoding the 8 bit memory word.
 - Or use bit mapped memory words directly for RGB.
 - Actually you cannot do this with the 6847.

- **You do have to code your own character generator.**
 - Use an external PROM or the internal ROM in an FPGA.



Quiz Review



- **Thursday, October 29, 2003**
 - 7:30 P.M. to 9:30 P.M.
 - Room 50 30 (Walker)
 - Closed Book

- **Venue**
 - Problem Sets 1- 5, Labs 1 - 3, Lectures 1 – 15
(Not including video or transmission lines)
 - VHDL entities and simple architectures
 - Details of syntax, e.g., placement of semicolons not important
 - Understand concurrent statements:
 - Assignment, instantiation, when else, with select when, process
 - Understand sequential statements (only within a process):
 - Assignment, if- then else, case when

- **General Topics**
 - Boolean Algebra, Elementary Logic Expressions
 - Combinational Logic, Karnaugh Maps, MSP, MPS
 - Latches and Edge Triggered Flip flops



Quiz Review- 2



■ More topics:

- **System Timing**
 - minimum period
 - minimum delay
 - clock skew
- **Counters**
- **Static RAMs**
- **FSMs**
 - state registers
 - combinational logic
 - VHDL implementation
- **Synchronization**
 - Short pulse catcher
 - Level to a pulse
- **Tristate**
 - What it is
 - VHDL implementation



Quiz Review- 3



■ More topics

- Avoiding glitches
 - Gating the clock
 - Registering the output

- Miscellaneous VHDL
 - Don't cares
 - Concatenation- &
 - Operands and result are of the same length.

- Hierarchical design
 - Start with one block.
 - Subdivide until blocks are simple.
 - Implement blocks in VHDL.
 - Test the blocks.
 - Wire the blocks together by instantiation.
 - Test the complete design.



■ More topics

□ Arithmetic

- Representation of negative numbers
 - sign – magnitude
 - Ones complement
 - Twos complement
- Adders
 - Ripple carry
 - Carry look ahead (idea – not details)
- Shift and add multiplier

□ Reconfigurable logic architecture

- Anti fuse versus RAM based FPGAs
- Combinational logic implementation
- Carry chain



Quiz Review- 5



■ More topics

□ Major – minor FSMs

- Simultaneous operation
- Can different clocks be used?

□ Analog building blocks

- Op Amps
- D to A
 - Glitching
 - R- Σ ladder
- A to D
 - Successive Approximation
 - Flash

□ Data Transmission

- Serial protocol
- Parallel protocol
- Transmission line
 - Termination – characteristic impedance
 - Crosstalk – ground alternate wires



Sample Lecture Slides



□ Perhaps these slides highlight important points.

2 13, 22, 23

3 4, 10, 16, 18, 20

4 8, 10, 11, 12, 16, 17, 23, 24, 26

5 3, 6, 8, 15, 16, 19, 20, 25, 27, 29

6 2, 11, 12, 13, 15, 17, 22, 23, 25

7 12, 23, 26

8 9, 11, 12

9 2, 27

10 2, 3, 4, 14, 16

14 2, 3, 4

12 1, 8, 24

13 1, 5, 16, 17, 18

15 3, 7, 8



Design Rules



- **Use hierarchical design.**
 - Small subsystems are easier to design and test.
- **Design for testability.**
 - Design subsystems so they will run alone.
 - Design test FSMs to be enabled by switches.
- **Do your logic design carefully, and first.**
 - Make up block and circuit diagrams.
 - Use names appropriate to the assertion levels.
- **Avoid problems from ‘glitches’.**
 - Gate delays can (and do) cause glitches.
 - Ensure a stable combinational output before it is sampled by CLK.
 - CLK, G, /PR, /CL inputs must NOT have glitches.
 - Create glitch free signals by
 - Registering outputs.
 - Gating the clock.



Design Rules- 2



- **Use the same clock edge for all edge triggered flip flops.**
 - Beware of clock skew.
 - Change inputs only (just) after the clock edge.
- **Avoid tristate bus contention.**
- **Be careful about Asynchronous events.**
 - Synchronize all external signals.
 - Consider pulse width carefully.
 - Should you have a short pulse or a sustained level?
- **Use memory properly**
 - Avoid high Z address to SRAM when CE is asserted.
 - Avoid address changes when WE is true.
 - Make sure your write pulse is glitch free.
 - Use a logic analyzer or a scope.



How to Make Your Project Work



- **Read (and heed) all of the handout.**
 - It is 'old' but all of it is good advice.
- **Sections that are particularly relevant are:**
 - **Wiring Errors**
 - **Care and Feeding of the Power Supply**
 - **Unused Inputs**
 - **Behavior of Ungrounded Parts**
 - **Tri- State Logic Signals**
 - **Handling CMOS Parts**
 - **Wire Routing**
 - **Clock Distribution**
 - **Gating the Clock**
 - **RAM Write Pulses**
 - **Synchronizer Errors**
 - **Testing Strategies**
 - **Driving High Current Devices**