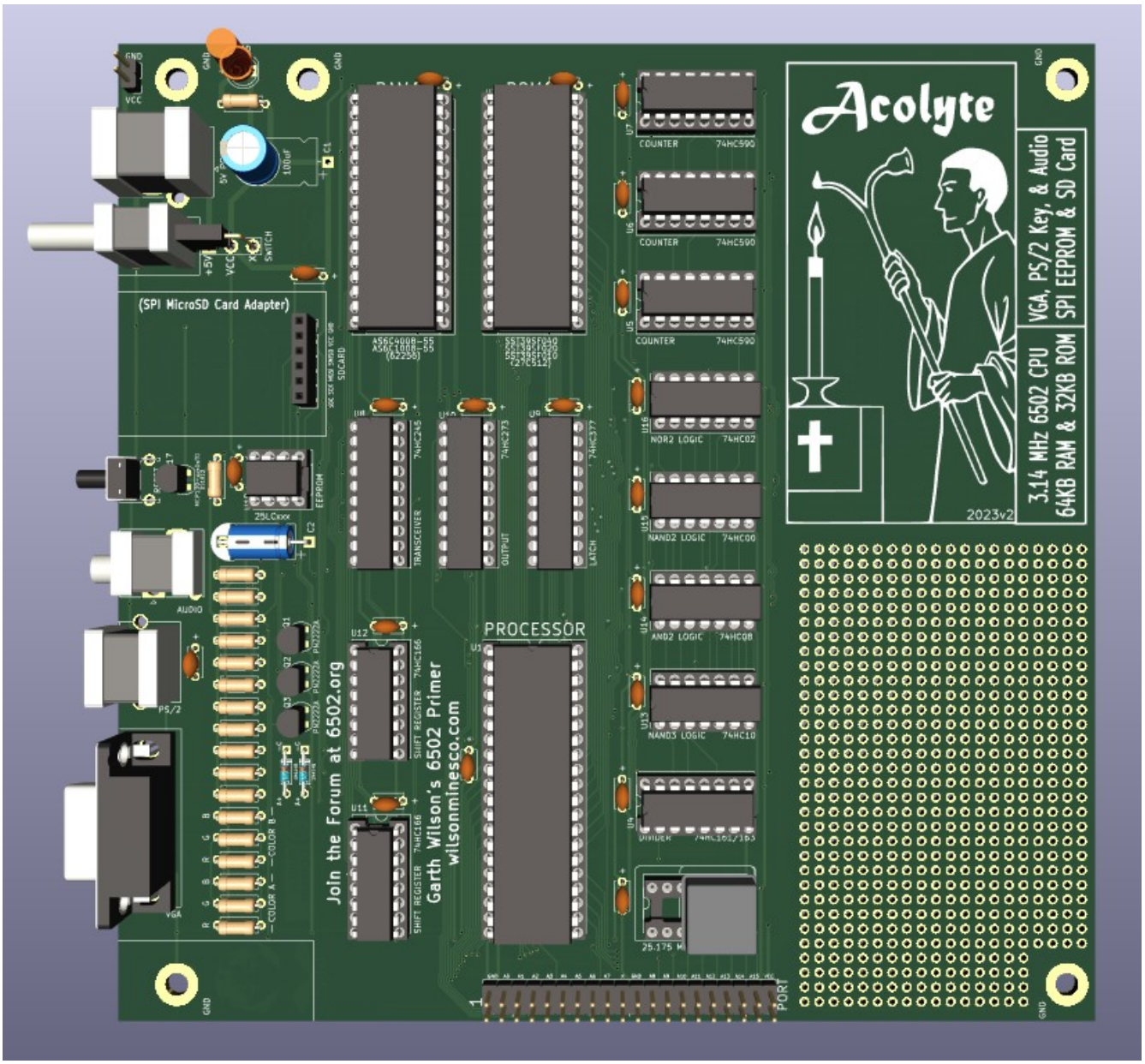
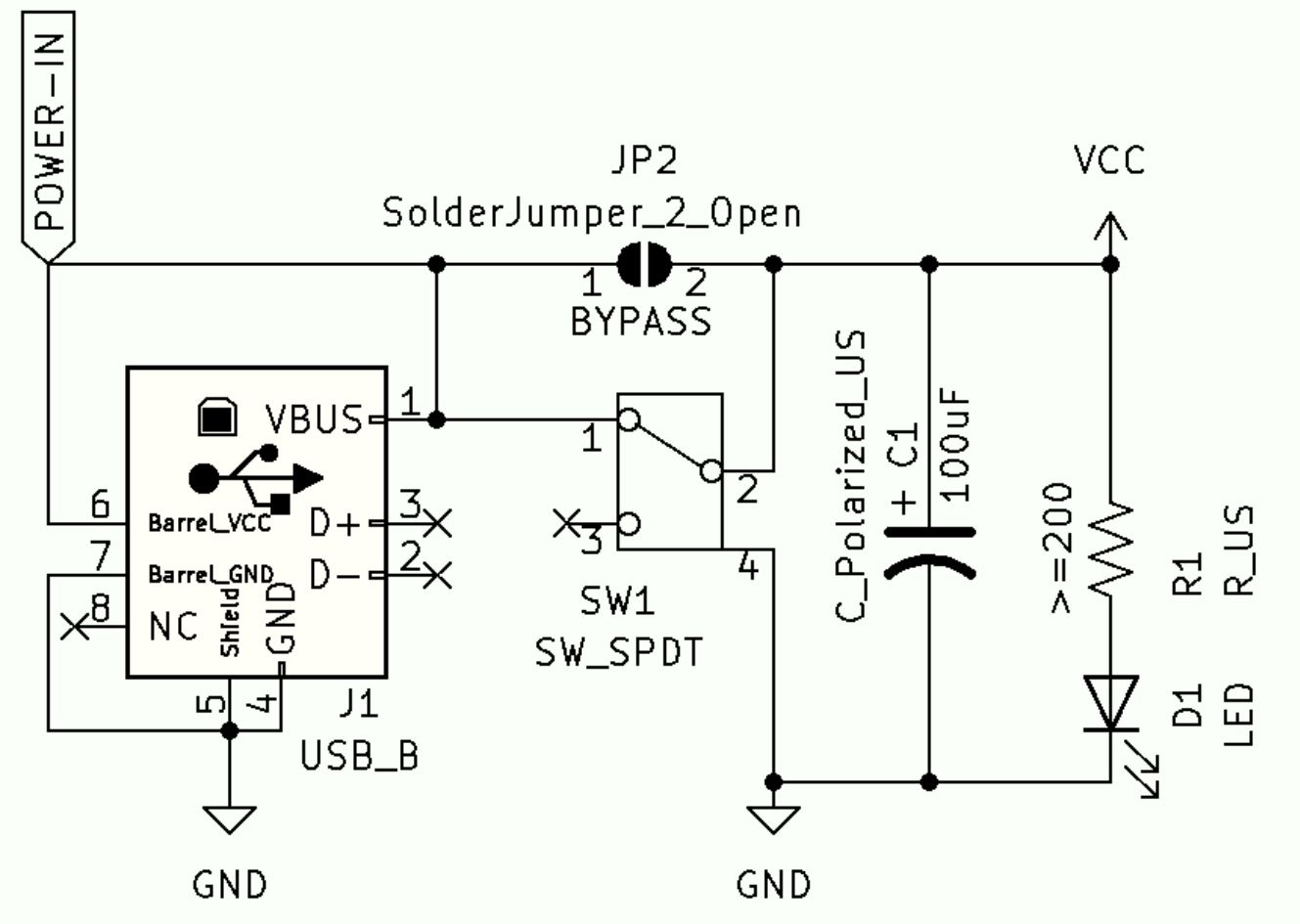
ACOLYTE COMPUTER



The Acolyte Computer is a “Homebrew” computer. It runs a 6502 at 3.14 MHz and has up to 64KB of RAM and 32KB of ROM available. Uses a PS/2 Keyboard, VGA, Audio, SPI EEPROM, and supports an SD Card.

POWER



The Acolyte Computer is powered by a 5V source, typically a USB wall charger. The J1 USB connector on the board is a USB-B type, often used with printers, scanners, and other peripherials. The footprint on the board allows for use of a generic barrel jack as an alternative.

5V power then goes to the SW1 switch making it convenient to turn the the Acolyte Computer on and off without needing to constantly unplug it's power source. If the switch is not wanted, connect the JP2 "bypass" jumper ends with a blob of solder.

On the board, any label of "+5V" is from the power source, while any label of "VCC" is power after the switch.

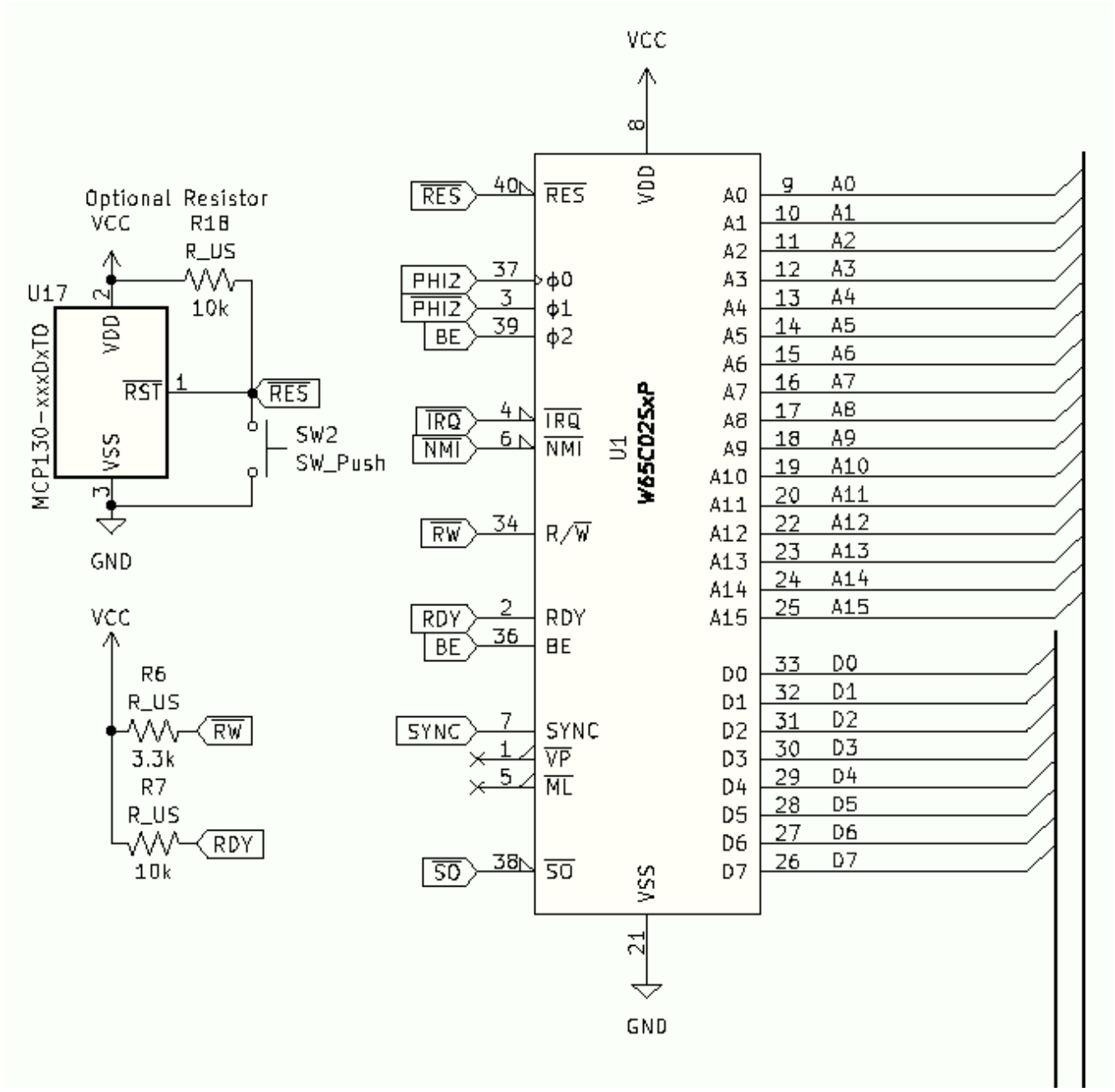
The C1 100uF decoupling capacitor helps regulate power fluctuations on the board. It is polarized, so installing it in the correct direction is absolutely critical. If flipped it could literally explode! The footprint on the board allows for either radial or axial versions of this capacitor.

The D1 LED light shines when the board is powered on. If it were directly connected to the 5V power source it would burn up, so the R1 resistor prevents that from happening. The smaller the resistance, the brighter the LED shines. It has a value of >=200 ohms, but typically a 1K ohm resistor works fine.

Components:

J1 - USB	USB-B Connector	
SW1 - SWITCH	SPDT Switch	
C1 - 100uF	100uF Polarized Capacitor	
D1 - LED	LED Light	
R1 - >=200Ω	1K ohm Resistor	

PROCESSOR



The Acolyte Computer uses the U1 W65C02S 8-bit microprocessor, in current production by WDC. This processor was used in the many famous computers in the early 1980's including the Apple II, Commodore 64, and the Nintendo Entertainment System.

Signal Descriptions:

A0 – A15	Address bus, capable of accessing 64KB of memory at a time (output).
D0 – D7	Data bus, capable of 256 numeric values (bi-directional).
/RES	Reset signal, resets the processor (input).
PHI2	Clock (input).
/PHI2	Inverted clock (output).
BE	Bus Enable, takes processor buses offline.
/IRQ	Maskable Interrupt, level triggered (input).
/NMI	Non-maskable Interrupt, edge triggered (input).
/RW	Read-Write signal, connected to memory (output).
RDY	Ready signal, halts the processor (input).
SYNC	Sync signal, tells when opcode is used (output).
/SO	Set Overflow, allows for fast polling (input).

The job of the processor is to access memory locations from RAM or ROM using A0-A15 . The data sent back and forth is on D0-D7. The processor requires the PHI2 clock signal to operate, which is 3.14 MHz on the Acolyte Computer.

In order to have a video display, the processor's address and data buses are taken offline half of the time. This is done with the BE signal. While PHI2 is low the video signals populate the address and data buses, and while PHI2 is high the processor is allowed to access RAM and ROM for normal operation.

Pull-up resistor R6 is connected to the /RW signal. This is here because the /RW line floats when BE is low. Although the glue logic takes care of any level of /RW during this time, a pull-up resistor stabilizes the signal. Another pull-up resistor R7 is on the RDY signal. Although RDY is not used on the Acolyte Computer, it could now be used through the expansion port.

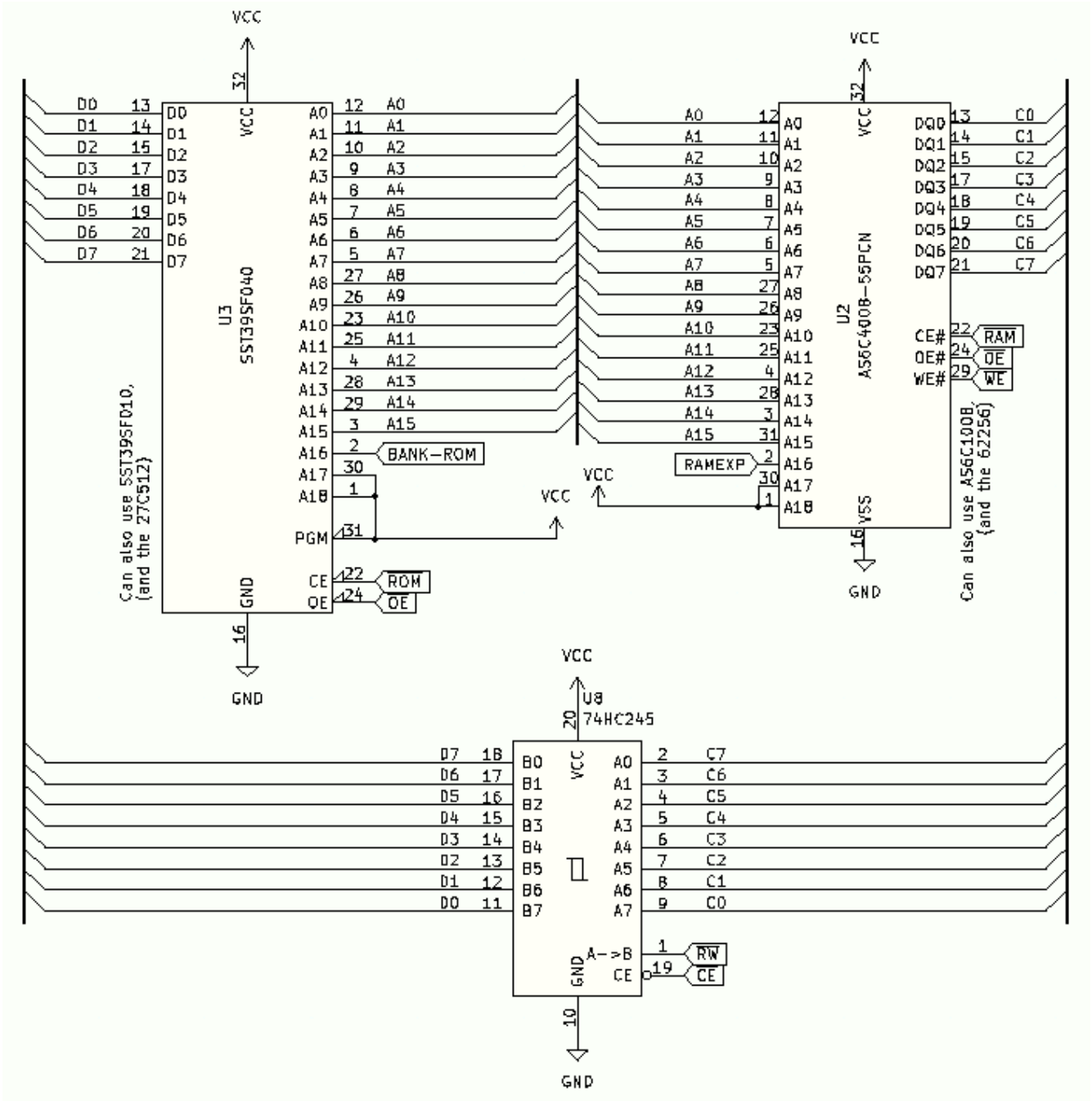
The Acolyte Computer uses many 'unconventional' methods with the 6502 processor. Using the processor's built-in /PHI2 inverter saves some glue logic. The BE signal is generated by the processor's PHI2-OUT to (in theory) delay the signal. The /IRQ and /NMI signals are almost directly connected to the PS/2 Keyboard, and the /SO line is the inverted SPI-MISO signal. Though not directly used on-board, the SYNC signal can be used to detect $_3 illegal opcodes for extra output bits.

The U17 Supervisory Reset Circuit will reset the processor after power-up. This is critical, and without this tiny IC the processor will glitch every time the power is turned on. The SW2 push button is for reseting the processor without having to turn the power off. The R18 pull-up resistor is not required if the Supervisory Reset Circuit is used but should still be installed.

Components:

U1 – PROCESSOR	W65C02S Microprocessor	
U17 – RESET	MCP130xxD Reset Circuit	
SW2 – RESET	SPST Reset Pushbutton	
R6 – 3.3KΩ	3.3K ohm Resistor	
R7 - 10KΩ	10K ohm Resistor	
R18 - 10KΩ	10K ohm Resistor (optional)	

MEMORY



The Acolyte Computer uses the U2 128KB+ SRAM chip for volatile memory, and the U3 128KB+ FlashROM chip for read-only memory. They share the address bus but the U8 Transceiver chip is capable of disconnecting their data buses.

Signal Descriptions:

A0 – A15	Shared address bus (input).
D0 – D7	ROM data bus, connected directly to processor (bi-directional).
C0 – C7	RAM data bus, connected through transceiver (bi-directional).
/RAM	RAM enable signal (input).
/ROM	ROM enable signal (input).
/OE	Output-Enable (input).
/WE	Write-Enable (input).
/RW	Read-Write (input).
/CE	Chip-Enable (input).
RAMEXP	RAM address expansion signal (input).
BANK-ROM	ROM address bank (input).

For the sake of versatility, many different versions of the RAM and ROM chips can be used interchangably. For example, the RAM could be an AS6C1008 or AS6C4008 without any changes in functionality. Similarly, the ROM could be an SST39SF040 or SST39SF010 without any changes to functionality.

Still further though, the RAM could be any version of the 32KB 62256 chip (given it is sufficiently fast enough), but it would then lose 16KB of available memory. This would disable BASIC on the Acolyte Computer, but all other functions will still work as expected. Similarly, the ROM could be any version of the 64KB 27C512 (given it is sufficiently fast enough), but would lose a bank of memory.

The reason for the Transceiver is to separate video sync and reset signals from color data. All video sync and reset signals are on the ROM at locations $0000-$837F, and all video color data are on the RAM at locations $0800-$7FFF.

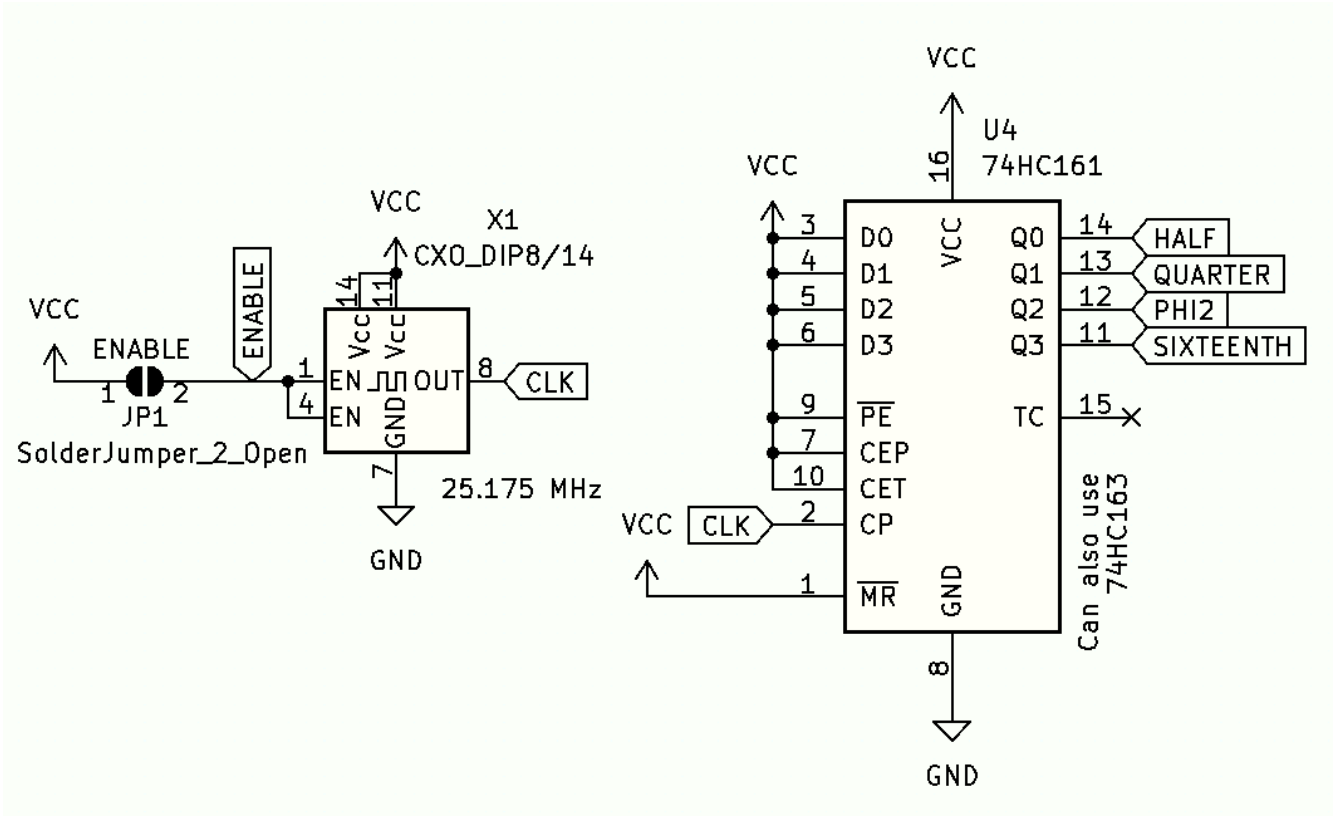
Volatile memory is on the RAM is accessable at locations $0000-$BFFF, with locations $8000-$BFFF being bankable. System read-only memory is on the ROM at locations $C000-$FFFF, but is banked to allow for more code.

All additional signals come from the “glue logic” portion of the Acolyte Computer.

Components:

U2 – RAM	AS6C1008 128KB SRAM	
U3 – ROM	SST39SF010 128KB FlashROM	
U8 – Transceiver	74HC245 Bi-Directional Bus Transceiver	

CLOCK



The Acolyte Computer has a master clock of 25.175 MHz from the X1 oscillator. This is used for the VGA video display signal. The processor only runs at 3.14 MHz, which is 1/8th of the original clock speed, so the clock is divided down using the U4 divider.

Signal Descriptions:

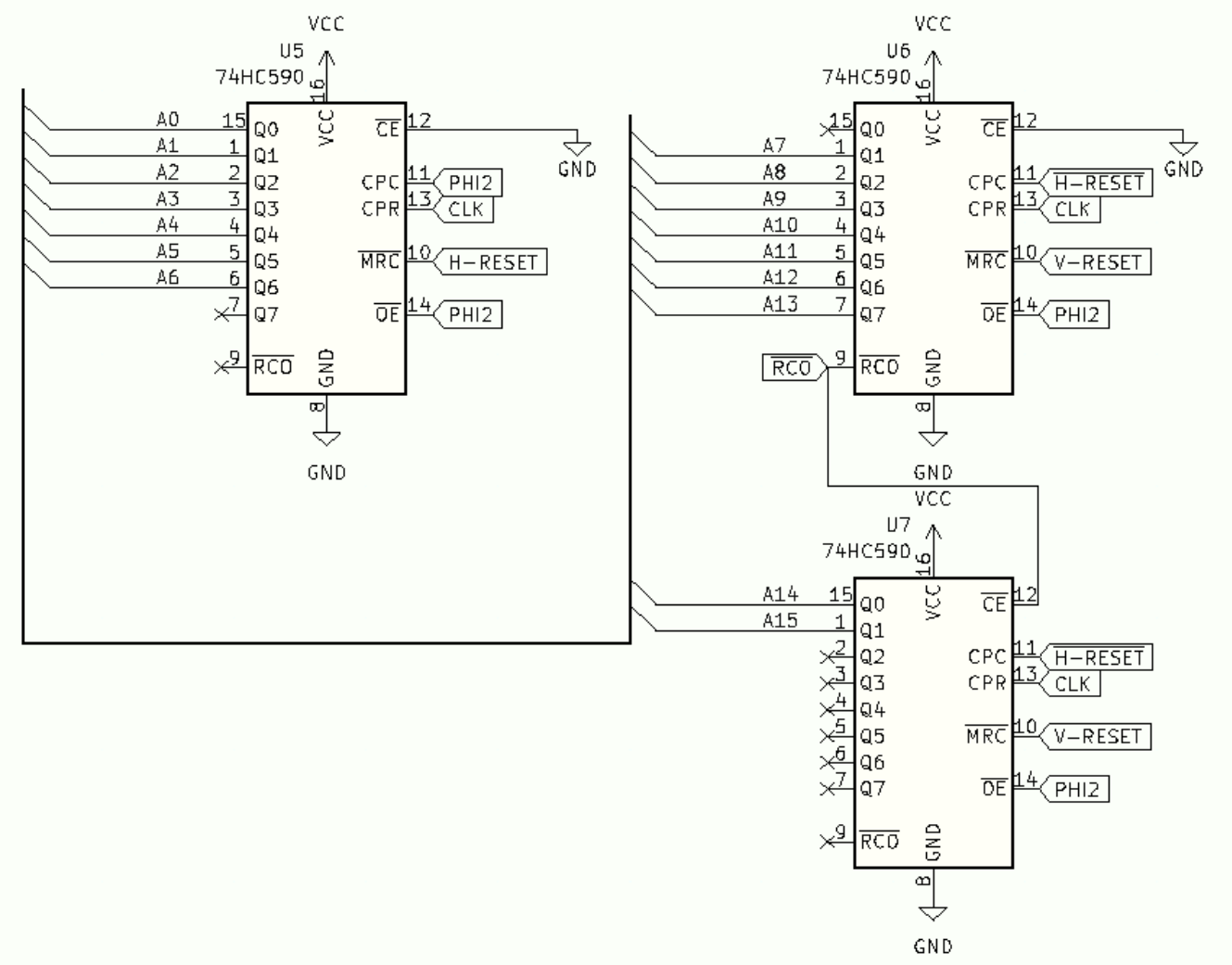
CLK	Master Clock at 25.175 MHz
HALF	Half of Master Clock at 12.59 MHz
QUARTER	Quarter of Master Clock at 6.29 MHz
PHI2	Eighth of Master Clock at 3.14 MHz, used by Processor
SIXTEENTH	Sixteenth of Master Clock at 1.57 MHz

The footprint for the Oscillator allows for either DIP-8 half-can or DIP-14 full-can oscillators. The JP1 “enable” jumper can be used if needed by the oscillator but is typically not required. The Divider could either be a 74HC161 or a 74HC163 4-bit counter chip.

Components:

U4 – DIVIDER	74HC161 or 74HC163 4-Bit Counter	
X1 - OSCILLATOR	25.175 MHz Can Oscillator (DIP-8 or DIP-14)	

COUNTERS



The Acolyte Computer uses the U5, U6, and U7 8-bit counters for accessing video sync and reset signals and color data for the video display.

Signal Descriptions:

A0 – A15	Address Bus (output).
CLK	Master Clock at 25.175 MHz (input).
PHI2	Eighth of Master Clock at 3.14 MHz (input).
H-RESET and /H-RESET	Horizontal Reset signals from ROM (input).
V-RESET	Vertical Reset signal from ROM (input).
/RCO	Carry signal used with cascading counters.

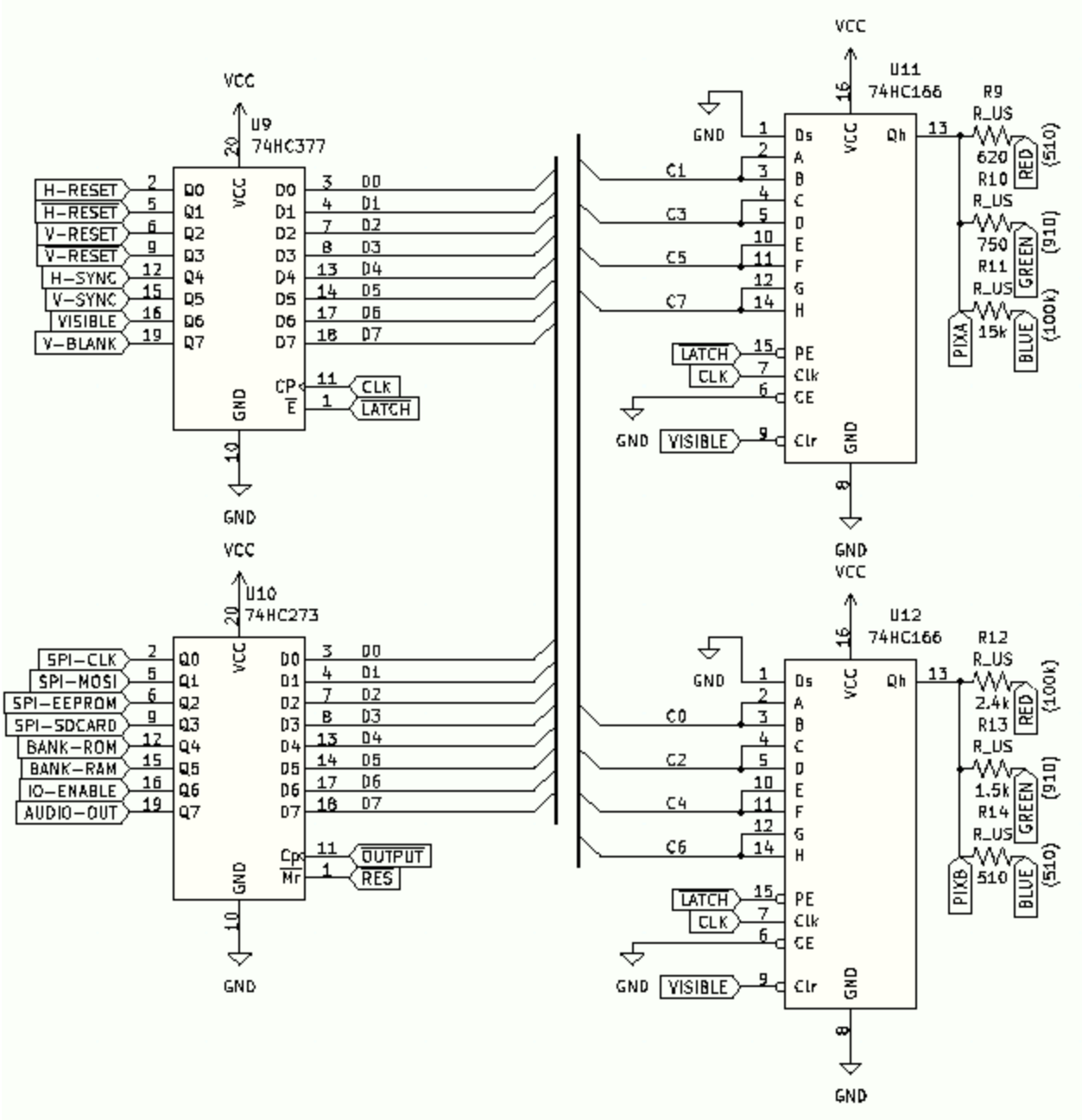
While PHI2 is low, these counters are online and access both the RAM and ROM. While PHI2 is high, these counters are offline, allowing the processor to access the address bus without conflict.

Once every PHI2 cycle the horizontal counter (U5) is incremented. When the location containing the H-RESET signal is accessed, the horizontal counter (U5) is reset, but the vertical counters (U6 and U7) are incremented. Finally when the location containing the V-RESET signal is accessed, the vertical counters (U6 and U7) are reset.

Components:

U4, U5, and U6 – COUNTER	74HC590 8-Bit Counters	

OUTPUT



The Acolyte Computer outputs various signals. The U9 latch holds all of the video display's sync and reset signals. The U10 latch holds various output bits for SPI interfacing, banking, and audio. The U11 and U12 shift registers hold the video display's color data, which goes through R9-R14 resistors.

Signal Descriptions:

D0 – D7	Data bus from ROM (input).
C0 – C7	Data bus from RAM (input).
CLK	Master Clock at 25.175 MHz (input).
/LATCH	Parallel Load Enable signal near end of PHI2 low (input).
/OUTPUT	Edge Triggered Load signal at end of PHI2 high (input).
/RES	Reset signal (input).
PIXA and PIXB	Color pixel data (output).
RED and GREEN and BLUE	Separated color data going to VGA connector (output).
H-RESET and /H-RESET	Horizontal Reset Signal (output).
V-RESET and /V-RESET	Vertical Reset Signal (output).
H-SYNC and V-SYNC	Horizontal and Vertical Sync Signals going to VGA connector (output).
VISIBLE	Visible color signal (output).
V-BLANK	Vertical blank signal (output).
SPI-CLK	SPI Clock signal (output).
SPI-MOSI	SPI Master-Out-Slave-In signal (output).
SPI-EEPROM	SPI Chip Enable for EEPROM (output).
SPI-SDCARD	SPI Chip Enable for SDCARD (output).
BANK-ROM	ROM Bank signal (output).
BANK-RAM	RAM Bank signal (output).
IO-ENABLE	I/O Enable for use with Expansion VIA (output).
AUDIO-OUT	Audio Output signal (output).

The U9 Latch and U11 and U12 Shift Registers all are directly used for the video display. The video sync and reset signals are programmed into the ROM, accessed by the 8-bit Counters (U5, U6, and U7), and then held by the U9 Latch. Video color data comes from the RAM, accessed by the same 8-bit counters, and then held by the U11 and U12 shift registers.

Each pulse of the 25.175 MHz master clock shifts the color data into view by the video display. Each color bit is duplicated on load, displaying each color for two pixels, making an effective resolution of 320 pixels across. Likewise the first output of the vertical counter (U6) is unconnected, duplicating each scanline, making an effective resolution of 240 pixels down.

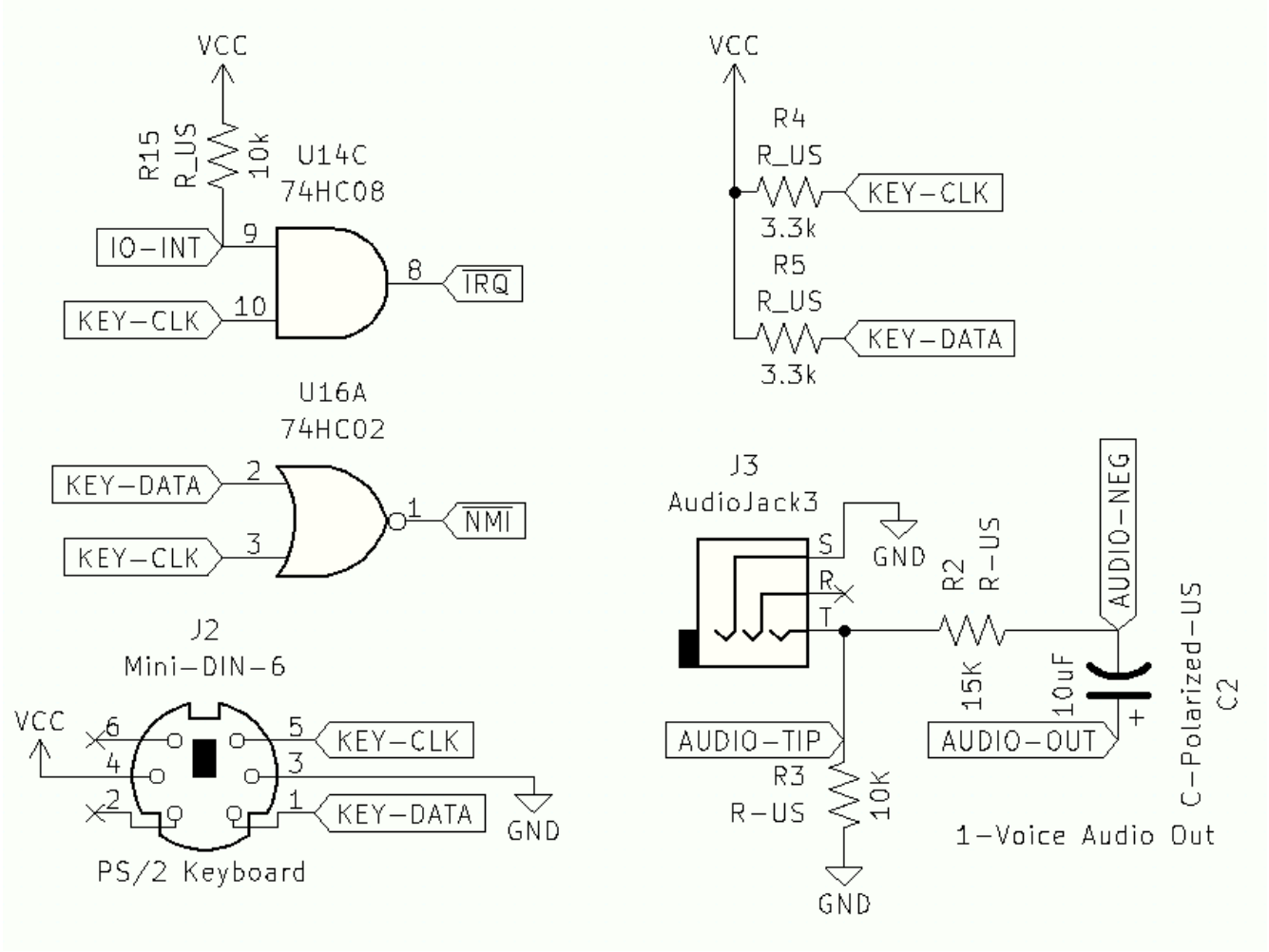
All SPI output bits come from the U10 latch, while SPI-MISO is inverted into the processor's /SO line. Banking the ROM allows for two separate sections of 16KB. Banking the RAM allows for two separate sections of 16KB, on top of the always visible 32KB (mostly dedicated to video color data).

The IO-ENABLE signal will disable the banked RAM from $8000-$BFFF, thus allowing that space to be used with a 6522 VIA or other I/O devices. Square wave audio is created by fluctuating the AUDIO-OUT signal at certain frequencies between 200 Hz and 2,000 Hz.

Components:

U9 – LATCH	74HC377 8-Bit Latch	
U10 – OUTPUT	74HC273 8-Bit Latch	
U11 – SHIFT REGISTER	74HC166 8-Bit Shift Register	
U12 – SHIFT REGISTER	74HC166 8-Bit Shift Register	
R9 – 620Ω	620 ohm Resistor	
R10 – 750Ω	750 ohm Resistor	
R11 – 15KΩ	15K ohm Resistor	
R12 – 2.4KΩ	2.4K ohm Resistor	
R13 – 1.5KΩ	1.5K ohm Resistor	
R14 – 510Ω	510 ohm Resistor	

KEYBOARD AND AUDIO



The Acolyte Computer maximizes useful Input/Output with minimum hardware. The J2 PS/2 keyboard gives access to over 100 buttons. The J3 audio jack can send square wave audio to external speakers or headphones.

Signal Descriptions:

KEY-CLK and KEY-DATA	PS/2 Keyboard Clock and Data signals (input).
IO-INT	I/O Interrupt from Expansion VIA (input).
/IRQ	Level-triggered Interrupt (output).
/NMI	Edge-triggered Interrupt (output).
AUDIO-OUT	Audio Output signal from U10 Latch (input).
AUDIO-NEG	Audio signal through C2 Capacitor.
AUDIO-TIP	Audio signal at Audio Jack tip (output).

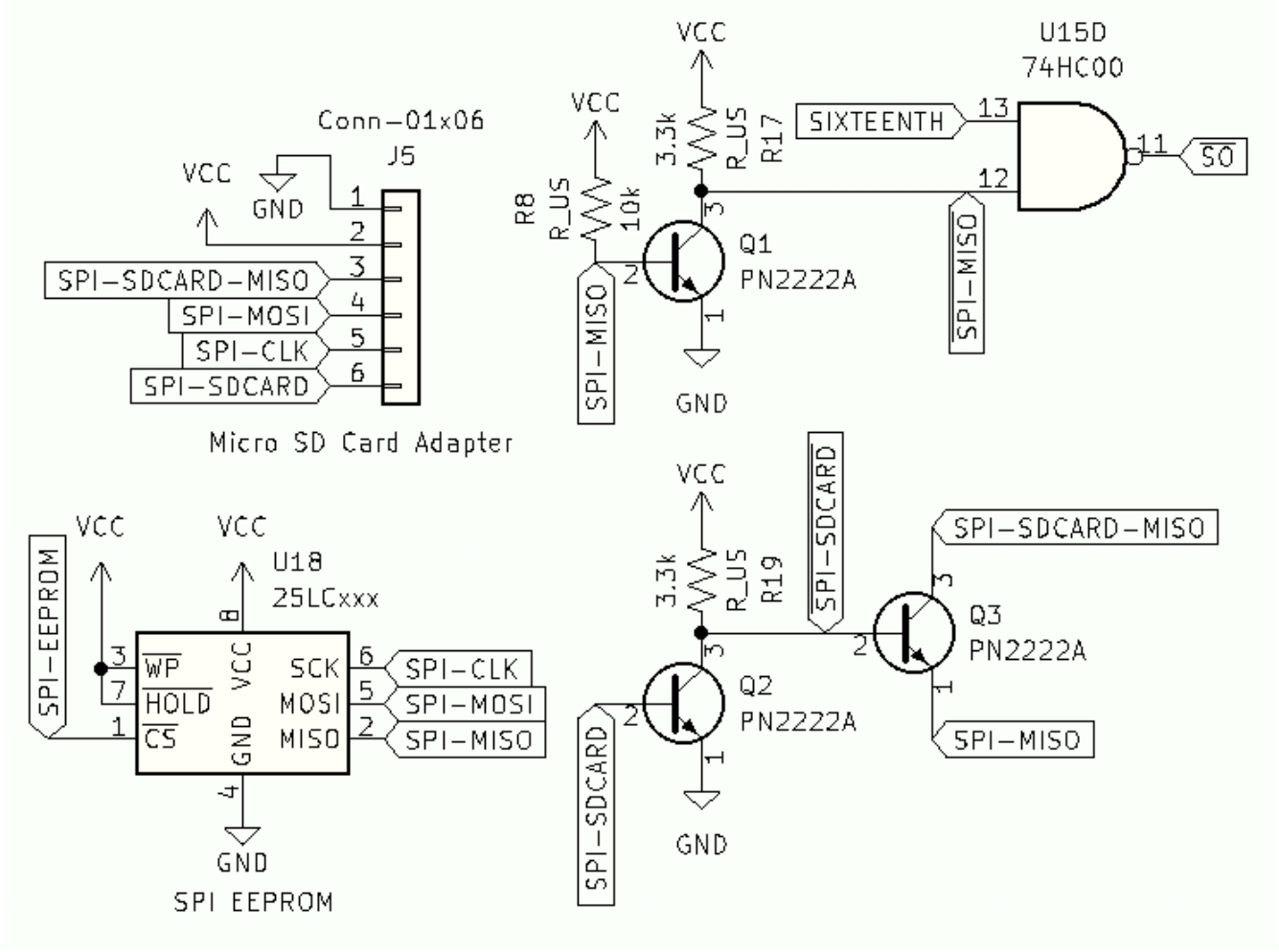
The PS/2 keyboard is connected to the processor interrupt lines through minimal logic. If IO-INT is left unconnected, /IRQ = KEY-CLK. In addition, /NMI = KEY-CLK nor KEY-DATA. This combination of logic allows for extraction of data from keyboards of various speeds though it also can delay program execution significantly if not programmed correctly.

Square wave audio is created by fluctuating the AUDIO-OUT signal. This causes the C2 Capacitor to create positive and negative voltages. The R2 and R3 Resistors set the signal between -1V and +1V as needed for external audio devices.

Components:

J2 – PS/2	PS/2 Keyboard Connector	
J3 – AUDIO	Audio Jack Connector	
C2 – 10uF	10uF Polarized Capacitor	
U14 and U15 – LOGIC	(See “Glue Logic”)	
R2 – 15KΩ	15K ohm Resistor	
R3 – 10KΩ	10K ohm Resistor	
R4 and R5 – 3.3KΩ	3.3K ohm Resistors	
R15 – 10KΩ	10K ohm Resistor	

SPI DEVICES



The Acolyte Computer uses SPI signals internally for additional memory storage. The U18 SPI EEPROM can store 8KB of memory even after power-down. The J5 connector leads to a SPI Micro SD Card Adapter, which can hold at least 2GB of data.

Signal Descriptions:

SPI-MISO	SPI Master-In-Slave-Out signal (input).
/SPI-MISO	Inverted SPI-MISO signal.
SIXTEENTH	Sixteenth of Master Clock at 1.57 MHz (input).
/SO	Set-Overflow signal (output).
SPI-xxx	SPI output pins from U10 Latch (input).
SPI-SDCARD and /SPI-SDCARD	Chip Enable signals for SD Card.
SPI-SDCARD-MISO	MISO coming from SD Card.

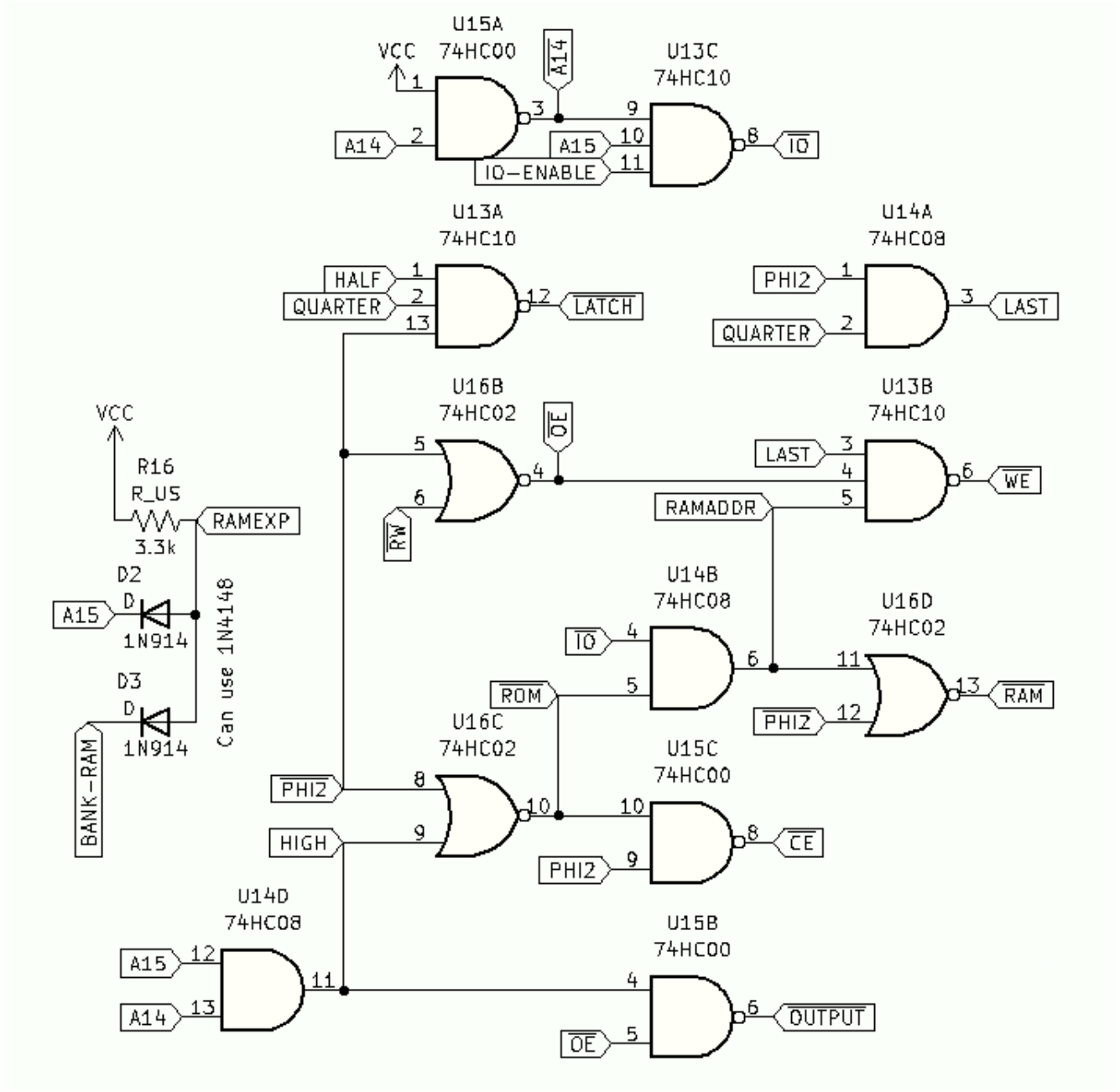
The SPI-MISO line goes through the Q1 Transistor to be inverted and then tied to a sixteenth of the master clock. This combination will toggle /SO when SPI-MISO is low thus triggering the overflow flag. Using a simple assembly code sequence of [CLV, NOP, BVS] would allow polling for MISO.

Because the SD Card does not release the MISO line after operation, Q2 and Q3 Transistors are used as a switch, cutting the SD Card's MISO line from the general MISO line when not in operation.

Components:

J5 – SDCARD	SPI Micro SD Card Adapter	
U18 – EEPROM	25LCxxx SPI EEPROM	
Q1, Q2, and Q3 – MISO	PN2222A Transistors	
U15 – LOGIC	(See "Glue Logic")	
R8 – 10KΩ	10K ohm Resistor	
R17 and R19 – 3.3KΩ	3.3K ohm Resistors	

GLUE LOGIC



The Acolyte Computer uses glue logic to tie together various signals so that they happen when needed. The U13 3xNAND, U14 2xAND, U15 2xNAND, and U16 2xNOR chips contain many logic gates to accomplish this goal.

Signal Descriptions:

PHI2 and /PHI2	Eighth of Master Clock at 3.14 MHz.
HALF and QUARTER	Half and Quarter of Master Clock.
A14 and /A14 and A15	Address lines.
IO-ENABLE	Output from U10.
/IO	Low when I/O is accessed.
/LATCH	Latch signal for U9, U11, and U12.
LAST	Clock pulse on last quarter of PHI2 cycle.
/RW	Read-Write from processor.
/OE	Output Enable, inverse of /RW.
/WE	Write Enable, inverse of /OE.
RAMADDR	Not ROM and not I/O.
/ROM and /RAM	Chip Enable for ROM and RAM.
/CE	Chip Enable for U8.
HIGH	Both A14 and A15 high.
/OUTPUT	Latch signal for U10.
RDY	Ready signal to processor.
RAMEXP	RAM bank.
BANK-RAM	Output from U10.

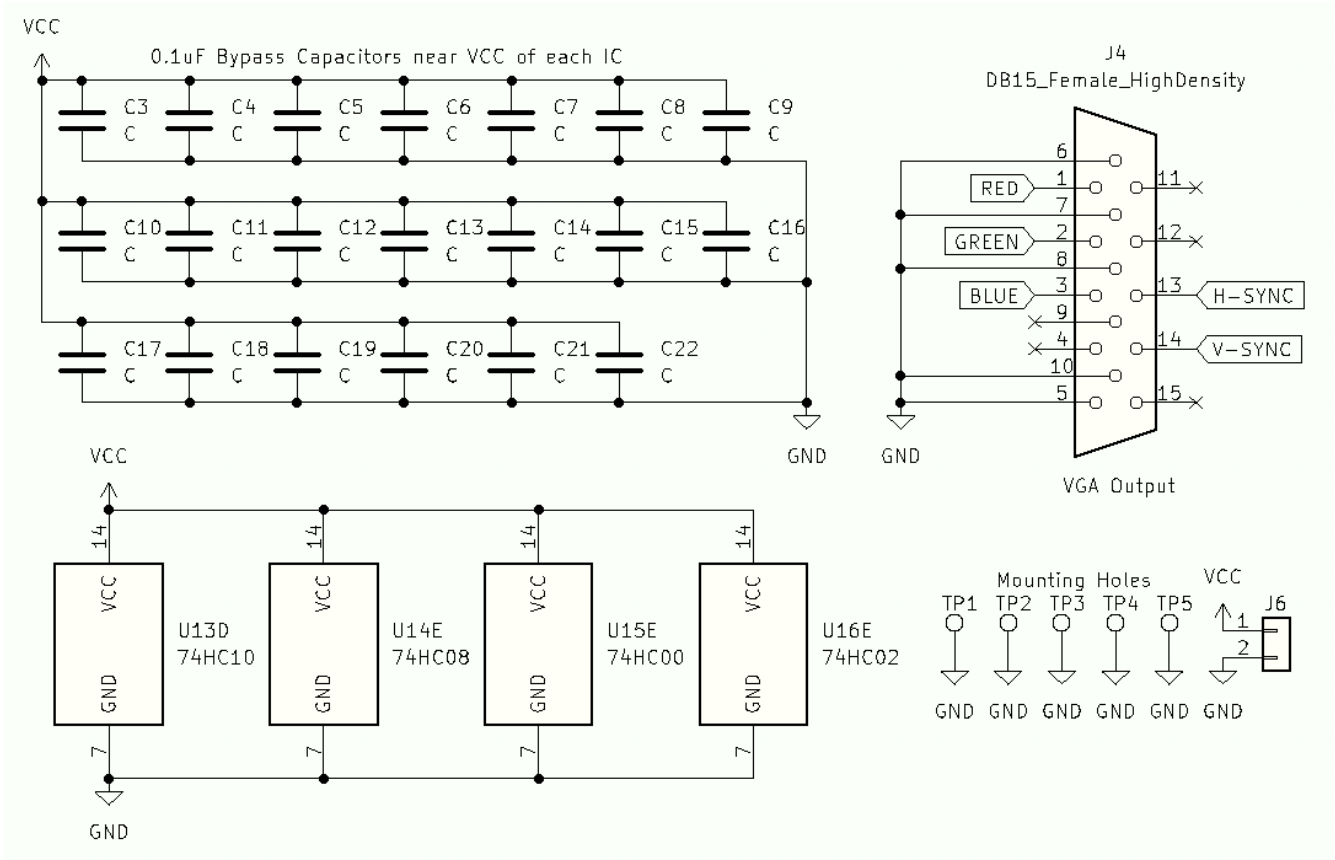
There is a lot going on here. The main thing to understand is that when PHI2 is low, data from RAM and ROM are for the video display. When PHI2 is high, the processor reads and writes to RAM and ROM normally.

The D2 and D3 diodes form “and” logic with A15 and BANK-RAM to create the RAMEXP signal. This both saves a logic gate and is an excuse to use some diodes on the board!

Components:

U13 – 3xNAND LOGIC	74HC10 Tri-NAND Logic	
U14 – 2xAND LOGIC	74HC08 Quad-AND Logic	
U15 – 2xNAND LOGIC	74HC00 Quad-NAND Logic	
U16 – 2xNOR LOGIC	74HC02 Quad-NOR Logic	
R16 – 3.3KΩ	3.3K ohm Resistor	
D2 and D3 – 1N914	1N914 or 1N4148 Signal Diodes	

MISC

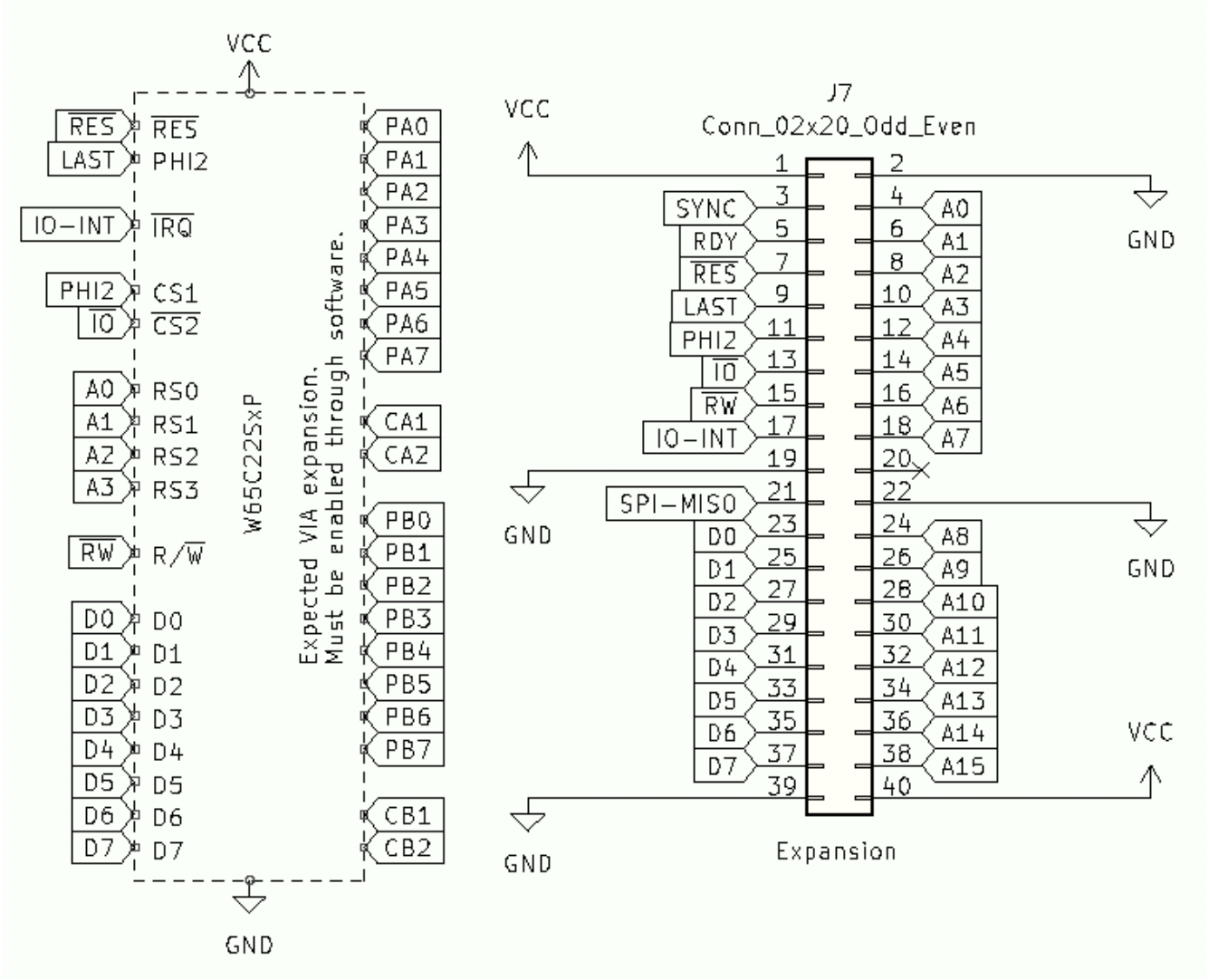


The Acolyte Computer has a lot of components. The J4 VGA connector has the RED, BLUE, GREEN, H-SYNC, and V-SYNC signals as an output to the video disiplay. The J6 pin header is a way to use VCC and GND from the board, or bypass the power circuit entirely. The TP1-TP5 test points are the mounting holes for the board. The U13-U16 chips were discussed in “glue logic”. And finally there are 20 Bypass Capacitors, all at 0.1uF. These need to be placed as close to each chip’s VCC and GND pins as possible.

Components:

J4 – VGA	DSUB15 VGA Connector	
C3 to C22 – 0.1uF	0.1uF Bypass Capacitors	

EXPANSION



The Acolyte Computer has the J7 expansion port which has many uses, one of which is to connect to a W65C22S VIA I/O Device. Another possible expansion is to use the SYNC and data bus to determine if a $_3 illegal opcode is being used, allowing for more output bits. SPI-MISO can be used as an additional input, but must be left floating when not in use.

Components:

J7 – PORT	2x20 Pin Header	

TIMING

The Acolyte Computer has many signals changing at various times for various reasons. The “glue logic” handles most of that, but here are more visual representations of what is going on.

Signal Descriptions:

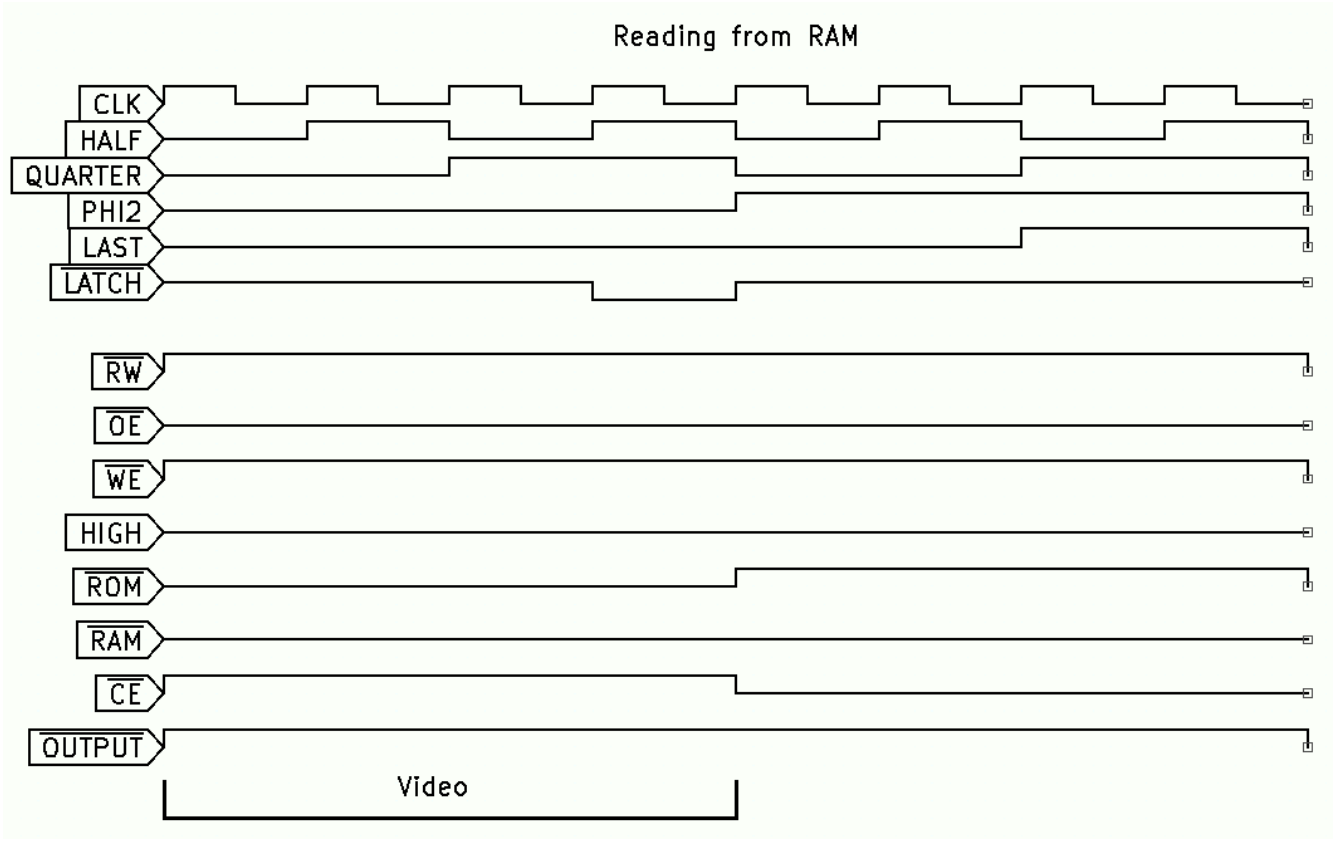
CLK	Master Clock at 25.175 MHz.
HALF	Half of Master Clock at 12.59 MHz.
QUARTER	Quarter of Master Clock at 6.29 MHz.
PHI2	Eighth of Master Clock at 3.14 MHz.
LAST	Clock pulse on last quarter of PHI2 cycle.
/LATCH	Latch signal for U9 signal latch, and U11 and U12 shift registers.
/RW	Read-Write (high = read, low = write).
/OE	Inverse of /RW, always low while PHI2 is low.
/WE	Inverse of /OE, always high while LAST is low.
HIGH	Both A14 and A15 are high, accessing $C000-$FFFF memory, can only happen while PHI2 is high.
/ROM	Chip Enable for ROM.
/RAM	Chip Enable for RAM.
/CE	Chip Enable for U8 transceiver.
/OUTPUT	Latch signal for U10 output latch.

In each of the the following diagrams, the first half of the PHI2 cycle is devoted to video output, which enables both the RAM for color data and the ROM for sync and reset signals. /RW, /WE, /CE, and /OUTPUT will always be high during this time, and /OE and HIGH will always be low during this time.

While PHI2 is low, /RW would be floating due to the BE signal on the processor going low. There is a 10K ohm pull-up resistor for safety, but the glue logic will never let a write happen while PHI2 is low as it always forces /OE low instead causing /WE to be high.

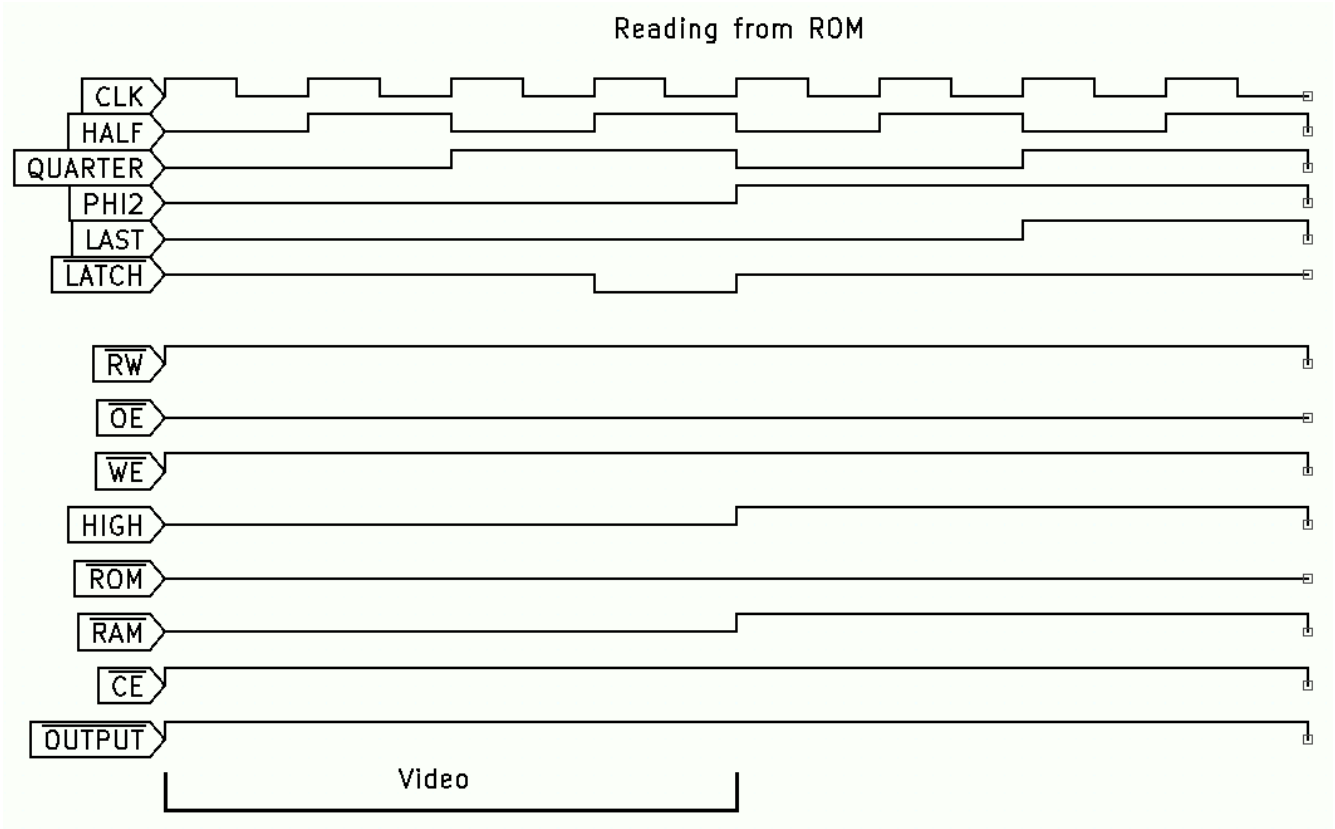
The /CE will always be high while PHI2 is low so that the data bus from the ROM carrying the sync and reset signals and the data bus from the RAM carrying the color data never interfere with one another.

There is also never a chance of the /OUTPUT signal going low while PHI2 is low because the video counters can never access the $C000-$FFFF portion of memory, only the processor can do that (besides the fact that you cannot write while PHI2 is low).

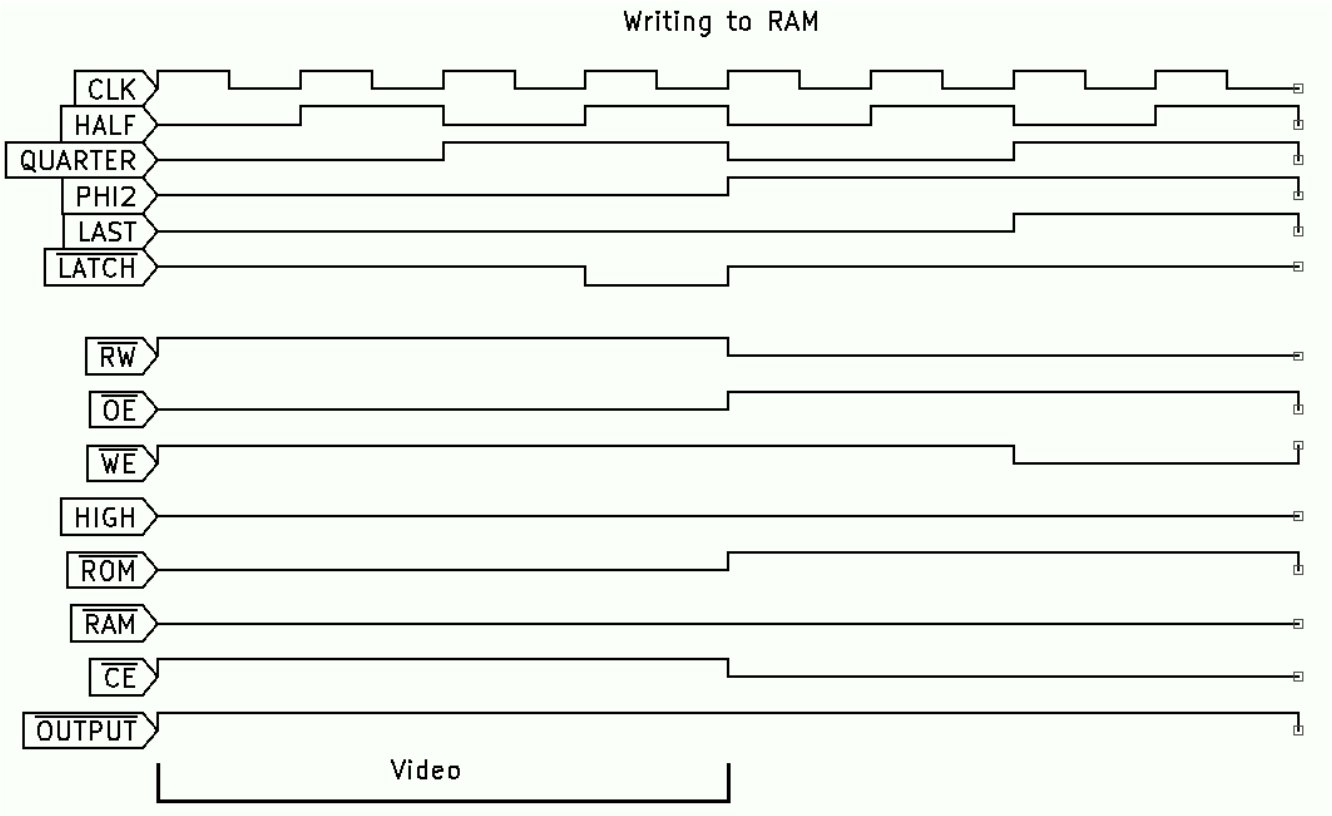


Reading from RAM keeps the /RW and /WE signals high, and the /OE signal low. /ROM is disabled but /RAM is kept enabled. /CE is enabled, allowing the processor's data bus connect to the RAM's data bus. /OUTPUT is kept high. Data is latched inside the processor on the falling edge of PHI2.

When the I/O space is enabled to overlap the RAM locations $8000-$BFFF, reading from those locations might keep the /RAM signal high, allowing for the I/O device to use the data bus freely. This goes for writing to RAM at those locations as well.

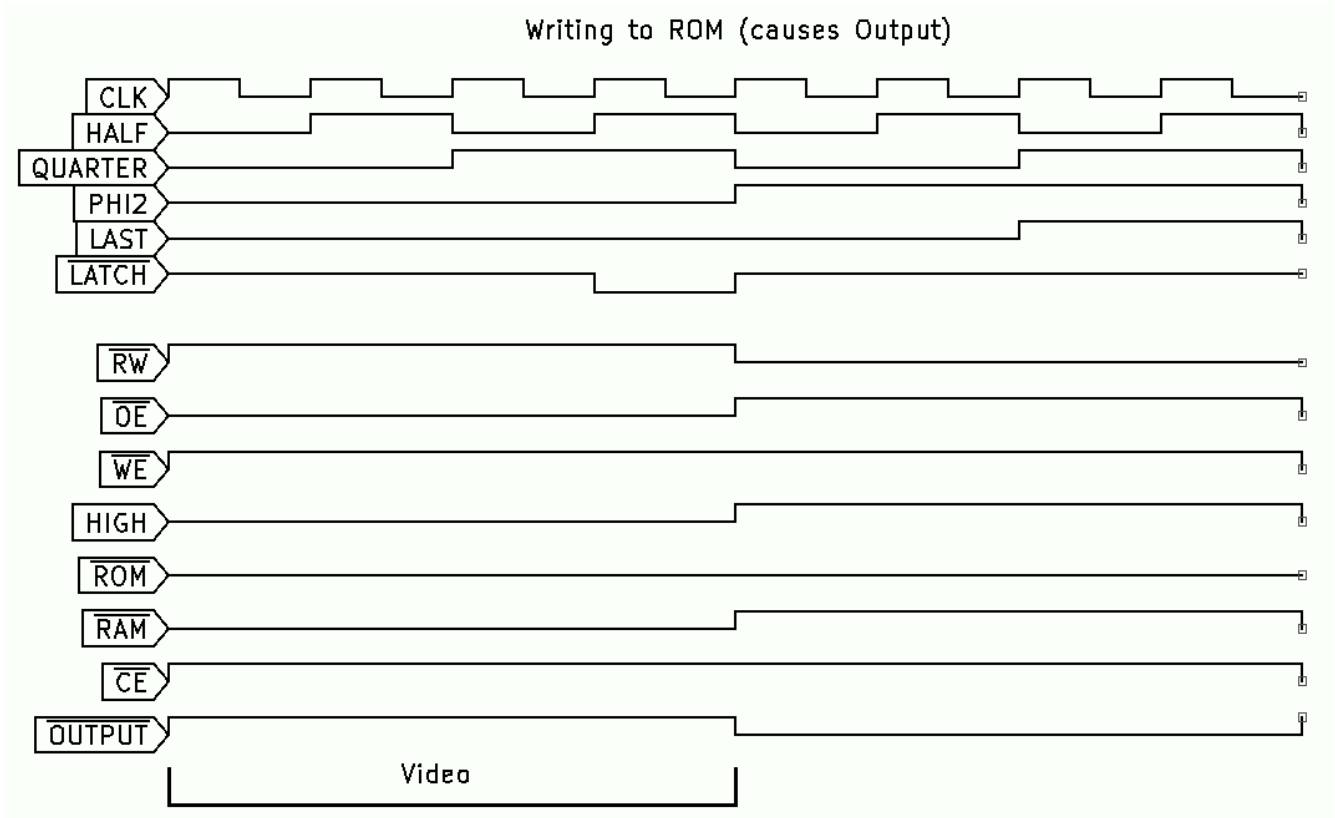


Reading from ROM keeps the /RW and /WE signals high, and the /OE signal low. /RAM is disabled but /ROM is kept enabled. This is done by HIGH going high. /CE and /OUTPUT are kept high. Data is latched inside the processor on the falling edge of PHI2.



Writing to RAM causes /RW to go low, and thus /OE to go high. /ROM is disabled but /RAM is kept enabled. /CE is enabled, allowing the processor's data bus connect to the RAM's data bus. /OUTPUT is kept high. When LAST goes high, /WE goes low, allowing for the RAM to be written to.
Having /WE enabled simply during PHI2 causes spurious writes due to the BE signal speed and RAM being very fast. Qualifying the /WE signal to the last quarter of the PHI2 cycle is very important!

When the I/O space is enabled to overlap the RAM locations $8000-$BFFF, writing to those locations might keep the /RAM signal high, allowing for the I/O device to use the data bus freely. This goes for reading from RAM at those locations as well.



Writing to ROM (for output bits) causes /RW to go low, and thus /OE to go high. /RAM is disabled but /ROM is kept enabled. This is done by HIGH going high. Although /ROM is enabled it cannot be accessed because the /OE signal is high. /CE is kept high. /OUTPUT goes low when PHI2 goes high, but the latching of the data bus for the output bits happens on the rising edge of /OUTPUT when PHI2 falls.

KEYBOARD

The Acolyte Computer uses an unconventional method to read the PS/2 keyboard, using both /IRQ and /NMI interrupt lines. As long as the I/O expansion device is disabled (by default in hardware), the interrupts will be as described.

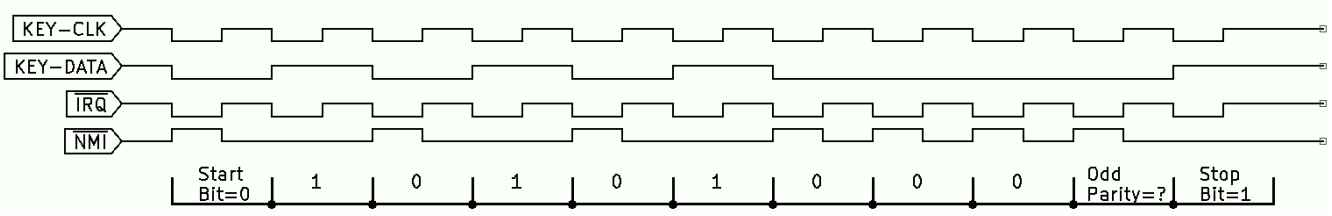
Signal Descriptions:

/IRQ – Maskable Interrupt, Level-Triggered	/IRQ = KEY-CLK
/NMI – Non-maskable Interrupt, Edge-Triggered	/NMI = KEY-CLK nor KEY-DATA
KEY-CLK	PS/2 Keyboard Clock signal
KEY-DATA	PS/2 Keyboard Data signal

The PS/2 keyboard sends signals on both the Clock and Data lines. The Clock line is normally high until a bit of data is being transmitted, causing the Clock line to go low. Shortly after the Clock line will go high again.

The Data line is normally high, and will change if the data bit is a zero. It changes before the Clock line falls, and stays at the designed level after the Clock line already goes high.

The diagram below shows the keyboard sending the “Q” key. Least significant bit first, hex value of $15, binary value of %00010101:



There is always a “start bit” sent first, and it is always a 0. Then eight data bits are sent. Then a “parity bit” is sent for error detection (though unused on the Acolyte Computer). Finally a “stop bit” is sent last, and it is always a 1.

PS/2 keyboards send signals at various speeds, between 10kHz and 17kHz. This variability in speed requires most setups to read the Data line on the falling edge of the Clock line. The Acolyte Computer instead reads the Data line on (essentially) the rising edge of the Clock line. The Acolyte Computer also automatically detects the speed of the keyboard (in software) using each start bit, making all PS/2 keyboards compatible.

The process to read the keyboard would be to enter a “interrupt subroutine” when /IRQ falls. On the start bit, the subroutine waits for /NMI to fall, and the time elapsed is stored for all following bits. On any other bit, the subroutine shifts the previous bit over and then waits for the stored time for /IRQ to rise. Meanwhile /NMI would have either fallen storing a 0 for the next bit, or would not have changed retaining a 1 for the next bit. If the bit read is the eight data bit, the key value is stored in a “buffer” for the running program to read.

Because the /IRQ is technically shared with the I/O interrupt, using both an I/O device that interrupts and the PS/2 keyboard at the same time is possible but not recommended.

Here are the IRQ and the NMI interrupt subroutines:

```
; 6502 running at 3.14 MHz,
; PS/2 keyboard running at 17 kHz
; That makes 184 to 314 cycles between signals.
; Half of each would be 92 or 157 cycles for low /IRQ.

; /IRQ = Keyboard-Clock
; /NMI = Keyboard-Data NOR Keyboard-Clock

vector_irq                  ; 7
     PHA                    ; 3
     LDA key_bit            ; 4
     ROR A                  ; 2
     ROR key_data           ; 2
     LDA #$FF               ; 2
     STA key_bit            ; 4
     DEC key_counter        ; 6
     BEQ vector_irq_store   ; 2
     BMI vector_irq_reset   ; 2
     LDA key_counter        ; 4
     CMP #$09               ; 2
     BEQ vector_irq_first   ; 2
     LDA key_speed          ; 4
     INC A                  ; 2
     INC A                  ; 2
     NOP                    ; 2
     JMP vector_irq_wait    ; 3, total = 61
vector_irq_reset            ; 1
     LDA #$0A               ; 2
     STA key_counter        ; 4
     LDA key_speed          ; 4
     STZ key_speed          ; 4
     INC A                  ; 2
     INC A                  ; 2
     JMP vector_irq_wait    ; 3, total = 62
vector_irq_store            ; 1
     PHX                    ; 3
     LDA key_data           ; 4
     LDX key_write          ; 4
     STA key_array,X        ; 5
     INC key_write          ; 6
     PLX                    ; 4
     LDA key_speed          ; 4, total = 72
```

```
vector_irq_wait
     DEC A                    ; 2x
     BNE vector_irq_wait      ; 3x
     PLA                      ; 4
     RTI                      ; 6, total = ?
vector_irq_first              ; 1
     NOP                      ; 2
     NOP                      ; 2
     NOP                      ; 2
     NOP                      ; 2
     NOP                      ; 2
     LDA #$00                 ; 2, total = 61
vector_irq_counter
     INC A                    ; 2x
     JMP vector_irq_counter   ; 3x

vector_nmi                    ; 7
     PHA                      ; 3
     LDA key_speed            ; 4
     BEQ vector_nmi_first     ; 2
     STZ key_bit              ; 4
     PLA                      ; 4
     RTI                      ; 6, total = 30
vector_nmi_first              ; 1
     PLA                      ; 4
     INC A                    ; 2
     STA key_speed            ; 4
     PLA ; may need PLP       ; 4
     PLA                      ; 4
     PLA                      ; 4
     STZ key_bit              ; 4
     PLA                      ; 4
     RTI                      ; 6, total = 53
```

VIDEO DISPLAY

The Acolyte Computer is essentially designed around the VGA video display. It uses the “industry standard” resolution of 640x480 pixels at 60Hz. To keep the processor’s speed managable, save video memory space, and to allow for color, the effective resolution for the Acolyte Computer is 320x240 pixels, though the display monitor will understand it as the 640x480 resolution.

Signal Descriptions:

RED	Red Color Component
GREEN	Green Color Component (only color if display is monochrome)
BLUE	Blue Color Component
H-SYNC	Horizontal Sync Signal
V-SYNC	Vertical Sync Signal

General Timing:

Screen Refresh Rate	60 Hz
Vertical Refresh	31.46875 kHz
Pixel Frequency	25.175 MHz

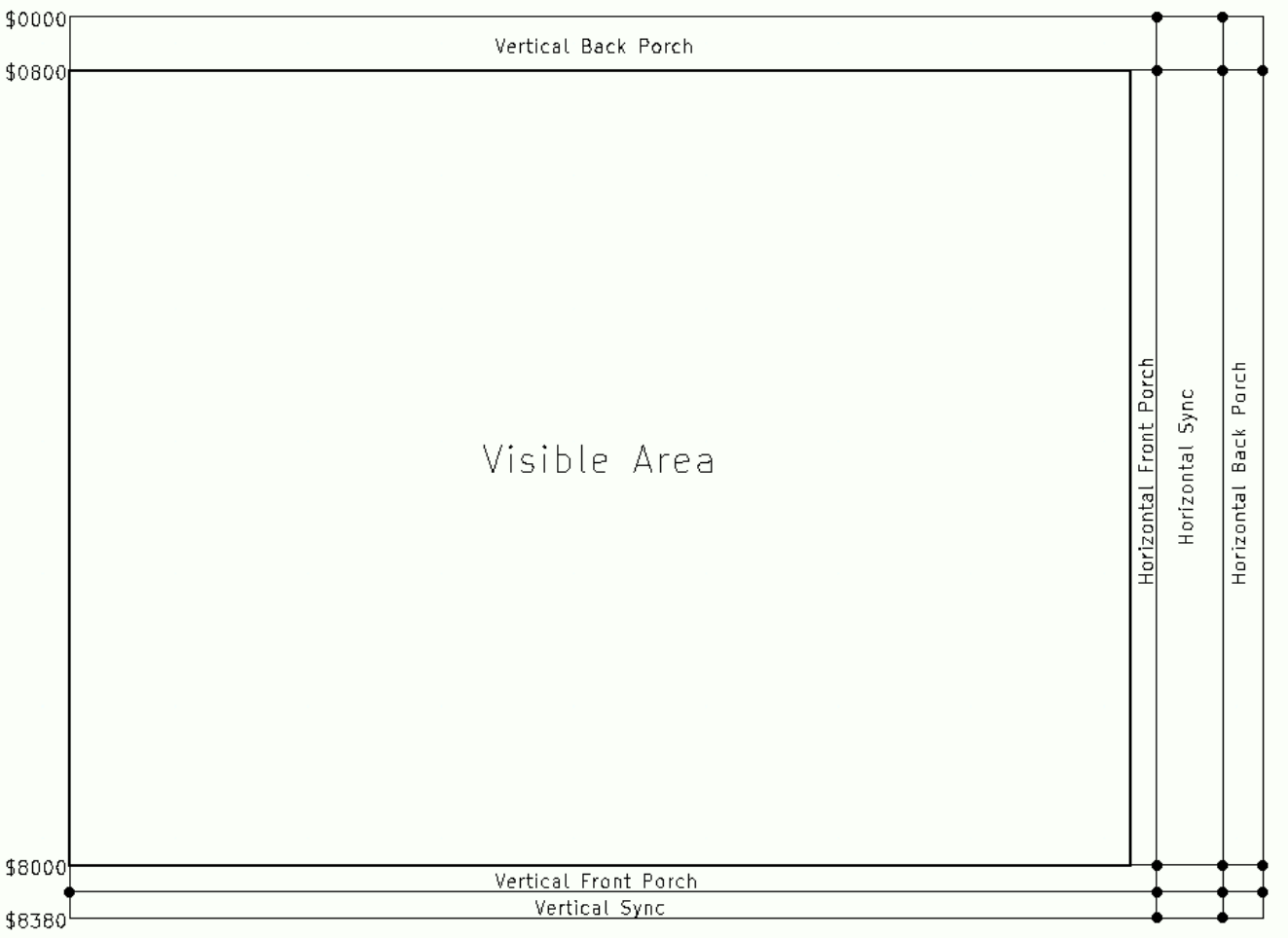
Horizontal Timing (Line):

Scanline Part	Pixels	Time [microseconds]
Visible Area	640	25.422045680238
Front Porch	16	0.63555114200596
Sync Pulse	96	3.8133068520357
Back Porch	48	1.9066534260179
Whole Line	800	31.777557100298

Vertical Timing (Frame):

Frame Part	Lines	Time [milliseconds]
Visible Area	480	15.253227408143
Front Porch	10	0.31777557100298
Sync Pulse	2	0.063555114200596
Back Porch	33	1.0486593843098
Whole Frame	525	16.683217477656

Even though most displays today are LCD or other digital screens, the way VGA displays is like how the old CRT monitors and televisions work. Imagine the beam racing from left to right on the screen, drawing specific pixels by turn on or off the beam and exact times. When the beam has reached the right side of the screen, a horizontal sync pulse happens, which brings the beam back to the left side of the screen but now shifted down one scanline. The beam then again draws going left to right. When the beam reaches the bottom of the screen, the vertical sync pulse happens, which brings the beam back to the top of the screen. A horizontal sync pulse at the same time also needs to happen so that the beam goes to the left side of the screen again! There are also some "porch" sections that are not visible on the screen.



The Acolyte Computer starts it's visible pixels at RAM memory location $0800, and ends them at $7FFF. Each scanline has 80 bytes of color data, so the top scanline would be in locations $0800-$084F. From $0850-$087F the color data is ignored entirely. This is called "overscan". The next scanline would be in locations $0880-$08CF, overscan from $08D0-$08FF. The third scanline would start at $0900. Thus, two scanlines are used for each 256 byte page in memory.

The horizontal and vertical sync signals are stored in ROM starting at memory location $0000 and ending at $837F. Other signals stored in ROM are the horizontal and vertical reset signals used for controlling the 74HC590 8-bit timers.

The exact signals would be:

Bit 0: H-RESET	Horizontal Reset Signal
Bit 1: /H-RESET	Inverse of Bit 0
Bit 2: V-RESET	Vertical Reset Signal
Bit 3: /V-RESET	Inverse of Bit 2
Bit 4: H-SYNC	Horizontal Sync Signal
Bit 5: V-SYNC	Vertical Sync Signal
Bit 6: VISIBLE	Visible Area Signal
Bit 7: V-BLANK	Vertical Blank Signal (unused)

The Acolyte Computer can display only 4 colors: Black, Blue, Orange, and White. The specific RGB values (by default) are:

BLACK	(0,0,0)
BLUE	(50,79,233)
ORANGE	(192,159,8)
WHITE	(242,238,241)

These specific colors were chosen for four reasons:

1) The two separate colors must add to “White”. Using RGB values, Orange + Blue = White.

2) The specific output power requirements from each 74HC166 shift register were considered to not stress the limitations of either chip.

3) Though other colors such as Magenta + Green = White, the Orange/Blue combination is more useful for displaying general natural scenery and is easy on the eyes.

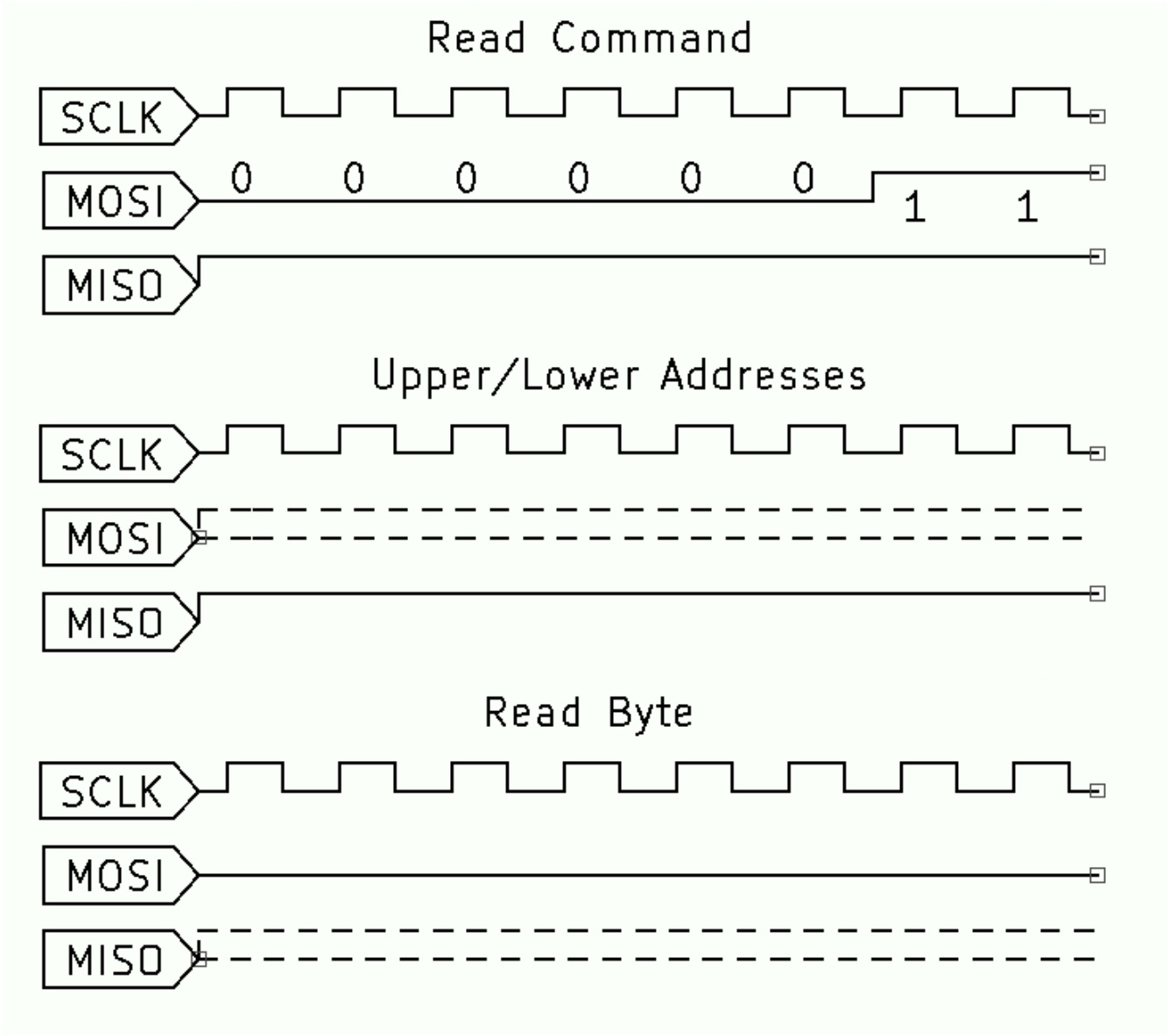
4) The particular shades of Orange and Blue were chosen so that a monochrome monitor using only the GREEN channel would still differentiate between the Orange and Blue values.

Each byte of memory in RAM in the visible area contains information for 4 pixels, thus 2 bits per pixel. Those two bits could be:

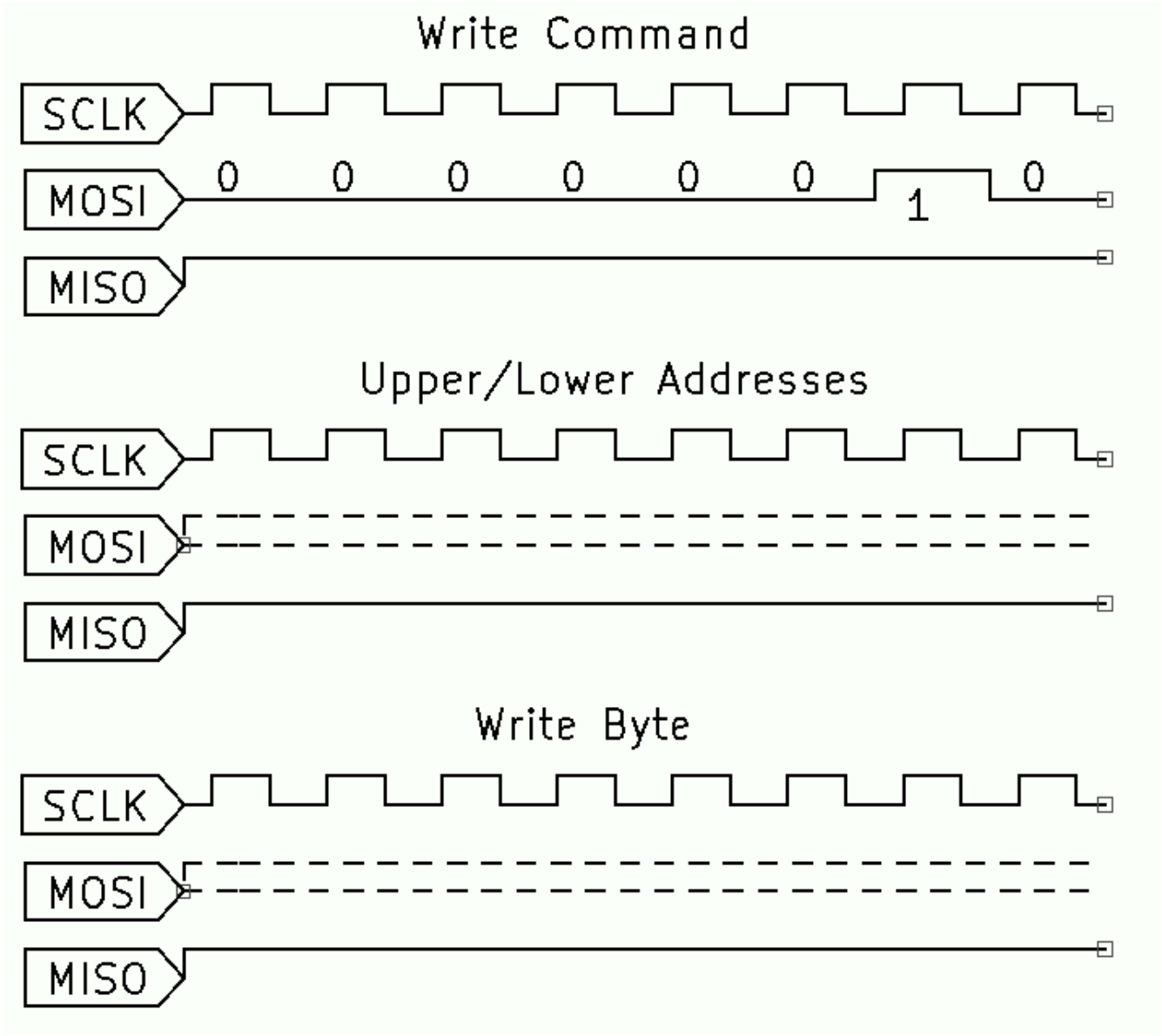
...00...	BLACK
...01...	BLUE
...10...	ORANGE
...11...	WHITE

SPI SIGNALS

The Acolyte Computer uses internal SPI functions for both the EEPROM and SD Card. Each share many pins, but how to interface them is quite different. In order to send commands to a specific SPI device, it's /CS pin must be held low. Let's look at the way to interface the EEPROM.



The commands are very simple, using only 8 bits at a time. Each bit is sent on the rising of the clock signal. First the "read command" of %00000011 is sent, then two address bytes are sent. Immediately after sending the last address bit, the MISO line no longer floats, but is driven by the EEPROM, containing the data in memory at the address specified.



Writing to the EEPROM is very similar. Before any of this write sequence can take place though, writing must be enabled. That is done by sending bytes: $01, $00, then $06. After that, we can write normaly. First, the “write command” of %00000010 is sent, then two address bytes are sent. Finally the data for the byte to be written is sent.

The EEPROM’s storage capabilities are designed to retain code, data, or even game states for when the Acolyte Computer is shut down. It does not hold as much data as the SD Card, but is quick and easy to access.

Interfacing the SD Card is a much more complicated process, with errors being very common on certain SPI Adapters. The Acolyte Computer also does not inherently support writing to the SD Card, and only reads binary images burned to the SD Card. That said, the storage capability of the SD Card is at minimum 2 GB, which is FAR larger than anything ever needed for the Acolyte Computer.

To interface the SD Card, it must first be initialized, and then it must be read from in 512 byte chunks. Within and between individual commands at various places require long delays and even clock pumping. Each command is still only 8 bits at a time.

In order to initialize the SD Card, the following byte sequence must be sent to the SD Card.

```
Send CMD0: $40, $00, $00, $00, $00, $95.

Receive: If $FF, try again. If $01, continue.  Else, error.

Send CMD8: $48, $00, $00, $01, $AA, $87.

Receive: If $FF, try again. If $01, continue.  Else, error.

Receive: 4-bytes, ignored.

Send CMD55: $77, $00, $00, $00, $00, $01.

Receive: If $FF, try again. If $01, continue.  Else, error.

Send CMD41: $69, $40, $00, $00, $00, $01.

Receive: If $FF, try again. If $01, send CMD55 and CMD41 again.
     If $00, continue.  Else, error.
```

Now that the SD Card is initialized, now we read a 512 byte block at a set location.

```
Send CMD17: $51, $00, $XX, $XX, $00, $01.

Receive: If $FF, try again. If $00, continue.  Else, error.

Receive: If $FF, try again. If $FE, continue.  Else, error.

Receive: 512-bytes, data.
```

The addresses used by the Acolyte Computer are at $XXXX, which point to a specific 256 byte block. Thus $0000 would read back the first 512 bytes on the SD Card, and $0010 would read back the second 512 bytes on the SD Card. This read sequence can be repeated indefinitely. If the EEPROM is interfaced, it is best practice to re-initialize the SD Card in case it misinterpretted stray signals as additional commands.

Although the SD Card is much more difficult to interface, the benefits outweigh the complexity. The SD Card itself can be connected to any modern computer and a binary file can be burned on it. This is how additional user programs, outside data, and even video games can be used by the Acolyte Computer.