Kansas Fest 93

Session: Peripheral Design

Presenter: Chuck Kelly / ProDev, Inc.
BUS TERMINATION

APPLE BUS IS NOT TERMINATED
ASYNCHRONOUS SIGNALS MAY NEED TO BE TERMINATED.

* This is any signal used without a clock qualifier. Address and data lines should always be used with a clock qualifier.

This termination method puts a minimal load on the bus but still provides noise reduction.

POWER FILTER

This diagram shows a power filter circuit with a 10μF capacitor and a 0.1μF disc capacitor. The filter is designed to reduce surges and high-frequency noise.

USE A BYPASS CAP AT EACH I.C.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>$2 \text{ MHz}$</th>
<th></th>
<th>$4 \text{ MHz}$</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Cycle Time</td>
<td>$t_{\text{CYC}}$</td>
<td>500 DC</td>
<td>250 DC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clock Pulse Width Low</td>
<td>$t_{\text{PWL}}$</td>
<td>0.24 10</td>
<td>0.12 10</td>
<td>10 µS</td>
<td></td>
</tr>
<tr>
<td>Clock Pulse Width High</td>
<td>$t_{\text{PWH}}$</td>
<td>240</td>
<td>120</td>
<td>nS</td>
<td></td>
</tr>
<tr>
<td>Fall Time, Rise Time</td>
<td>$t_{\text{f,fr}}$</td>
<td>- 10</td>
<td>- 10</td>
<td>nS</td>
<td></td>
</tr>
<tr>
<td>A0-A15 Hold Time</td>
<td>$t_{\text{AH}}$</td>
<td>10 - 10</td>
<td></td>
<td>nS</td>
<td></td>
</tr>
<tr>
<td>A0-A15 Setup Time</td>
<td>$t_{\text{ADS}}$</td>
<td>- 100 - 75</td>
<td></td>
<td>nS</td>
<td></td>
</tr>
<tr>
<td>BA0-BA7 Hold Time</td>
<td>$t_{\text{BH}}$</td>
<td>10 - 10</td>
<td></td>
<td>nS</td>
<td></td>
</tr>
<tr>
<td>BA0-BA7 Setup Time</td>
<td>$t_{\text{BS}}$</td>
<td>- 100 - 90</td>
<td></td>
<td>nS</td>
<td></td>
</tr>
<tr>
<td>Access Time</td>
<td>$t_{\text{ACC}}$</td>
<td>365 - 130</td>
<td></td>
<td>nS</td>
<td></td>
</tr>
<tr>
<td>Read Data Hold Time</td>
<td>$t_{\text{DHR}}$</td>
<td>10 - 10</td>
<td></td>
<td>nS</td>
<td></td>
</tr>
<tr>
<td>Read Data Setup Time</td>
<td>$t_{\text{DSR}}$</td>
<td>40 - 30</td>
<td></td>
<td>nS</td>
<td></td>
</tr>
<tr>
<td>Write Data Delay Time</td>
<td>$t_{\text{MD}}$</td>
<td>- 100 - 70</td>
<td></td>
<td>nS</td>
<td></td>
</tr>
<tr>
<td>Write Data Hold Time</td>
<td>$t_{\text{DHW}}$</td>
<td>10 - 10</td>
<td></td>
<td>nS</td>
<td></td>
</tr>
<tr>
<td>Processor Control Setup Time</td>
<td>$t_{\text{PCS}}$</td>
<td>40 - 30</td>
<td></td>
<td>nS</td>
<td></td>
</tr>
<tr>
<td>Processor Control Hold Time</td>
<td>$t_{\text{PCH}}$</td>
<td>10 - 10</td>
<td></td>
<td>nS</td>
<td></td>
</tr>
<tr>
<td>E,MX Output Hold Time</td>
<td>$t_{\text{EH}}$</td>
<td>10 - 10</td>
<td></td>
<td>nS</td>
<td></td>
</tr>
<tr>
<td>E,MX Output Setup Time</td>
<td>$t_{\text{ES}}$</td>
<td>50 - 50</td>
<td></td>
<td>nS</td>
<td></td>
</tr>
<tr>
<td>Capacitive Load (Address, Data, R/W)</td>
<td>$C_{\text{EXT}}$</td>
<td>- 100 - 100</td>
<td></td>
<td>pF</td>
<td></td>
</tr>
<tr>
<td>BE to High Impedance State</td>
<td>$t_{\text{BH}}$</td>
<td>- 30 - 30</td>
<td></td>
<td>nS</td>
<td></td>
</tr>
<tr>
<td>BE to Valid Data</td>
<td>$t_{\text{BDV}}$</td>
<td>- 30 - 30</td>
<td></td>
<td>nS</td>
<td></td>
</tr>
</tbody>
</table>

**AC Characteristics (W65C816)**

![Timing Diagram (W65C816)](attachment: timing_diagram.png)
The expansion slots

The figure below is a diagram of one of the seven expansion slots from an Apple II GS computer. The signals present on this connector are essentially the same as those present on every Apple II computer ever built.

In order to accommodate expansion cards built for the ][+ and //e the ][GS bus operates at the same clock speed as the older machines.

You can find a complete description of the signals present on the expansion connector in the Apple II GS hardware reference manual.

![Diagram of expansion slot](image)

**Signal Descriptions**

/IOSEL  Goes low when a peripheral cards $Cnxx$ space is active.

/IOSTRB  Goes low when $C800$ - $CFFF$ is accessed.

/DEVSEL  Goes low when a peripheral cards $C0nx$ space is active, where $n$ is the connector number plus 8.
NOTE! THESE DIMENSIONS EXCEED APPLE GS APPLE SPECS. SEE TECH NOTE #28.

NOTE 1 RADIUS OF CORNERS 0.06 UNLESS OTHERWISE INDICATED.
NOTE 2 ALL HOLE SIZES GIVEN ARE AFTER PLATING.
NOTE 3 CONNECTOR FINGERS ARE GOLD OVER NICKEL PLATING.
NOTE 4 MATERIAL FR4 0.062 THICK

HOLE CHART

<table>
<thead>
<tr>
<th>LABEL</th>
<th>SIZE</th>
<th>QUANTITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>0.037&quot;</td>
<td>542</td>
</tr>
</tbody>
</table>

BEVEL CORNERS 30 DEG TYP

TOLERANCES

<table>
<thead>
<tr>
<th>Tolerance</th>
<th>1/16</th>
<th>1/32</th>
<th>1/64</th>
<th>1/128</th>
<th>1/256</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fractional</td>
<td>1/16</td>
<td>1/32</td>
<td>1/64</td>
<td>1/128</td>
<td>1/256</td>
</tr>
<tr>
<td>Decimal</td>
<td>0.0625</td>
<td>0.03125</td>
<td>0.015625</td>
<td>0.0078125</td>
<td>0.00390625</td>
</tr>
</tbody>
</table>

PRODEV, INC.
MONROE, MICHIGAN

DDT rev A
**COMDOFF - TURN EXT DISPLAY OFF**

S_COMDOFF = )segnum ;segment number of this routine

COMDOFF BIT OFFFLAG ;DISPLAY ALREADY OFF ?

BMI GETCOMO ;IF YES

JSR TRANSFR0 ;RESTORE DISPLAY SWITCHES

DFB RESTTEXTC ;code

LDA #$80

STA OFFFLAG ;DON'T DISPLAY TO SCREEN

*----------------------------------------*

GETCOMO - do <CR> & wait for user command

*----------------------------------------

GETCOMO

PEA GETCOMCR-1 ;address of command

PEA S_GETCOM ;segment # of command

JMP JUMPSEGO ;goto command in other segment

**GLOBAL SUBROUTINES IN THIS SEGMENT *****

***** MAXIMUM OF 32 *****

***** CODE BYTES ARE EQUATED AS FOLLOWS

* BITS 0-2 = SEGMENT NUMBER $0 THRU $7 OF SUBROUTINE
* BITS 3-7 = NUMBER OF SUBROUTINE IN SUBTABL

SUBTABL0

LDAINDYC EQU *-SUBTABL0*4+0+$100

DA LDAINDY-1

STAINDC EQU *-SUBTABL0*4+0+$100

DA STAINDY-1

STEP1C EQU *-SUBTABL0*4+0+$100

DA STEP1-1

EXECUTEC EQU *-SUBTABL0*4+0+$100

DA EXECUTE-1

WSTKRES EQU *-SUBTABL0*4+0+$100

DA WSTKRES-1

AX2S1_AC EQU *-SUBTABL0*4+0+$100

DA AX2S1_A-1

************** SEGMENT CROSSOVER AREA **************

LST ON

S0END = $E0CF91-*

do nolist

LST OFF

fin

ERR *-$E0CF91
******** SAVE THE ACC, X, Y AND P REGISTERS *******
* Returns with MX = 11, saves registers

SAVEAXPO
  PHP          ;SAVE STATUS
  MX16
  STX XSAVESEG ;save 16 bits
  STY YSAVESEG ;save 16 bits
  STA ASAVESEG ;save 16 bits
  MX8
  PLA          ;GET STATUS
  STA PSAVESEG ;SAVE
  RTS

****** RESTORE THE ACC, X, Y AND P REGISTERS ******
* restores registers

RESTAXPO
  MEMORY8
  LDA PSAVESEG
  PHA
  MX16
  LDX XSAVESEG
  LDY YSAVESEG
  LDA ASAVESEG
  PLP

*** THIS RTS IS USED BY THE FINDSLOT ROUTINE ***
PUTSLOT RTS
  MX %11

*----------------------------------------*
* Do a direct transfer to other segments

JUMPSEGO
  JSR SAVEAXPO
  LDY SLOTNO
  PLA ;pull junk byte from dest. seg
  PLA ;get destination segment
  STA SEGMBASE,Y ;the next inst' will be in new seg
  JSR RESTAXPO ;restore after xfer from other seg
  RTS ;pull destination address from stack

* TRANSFER TO OTHER SEGMENTS
* Upper byte of X index is zeroed

TRANSFR0
  JSR SAVEAXPO
  MEMORY16
  PLA ;get return address from stack
  INC ;inc to point at code byte & for RTS
  PHA
  MEMORY8
  LDA #0      ;CURRENT SEG #
  PHA
LOY #0
LDA (2,S),Y ;GET CODE BYTE
PHA ;SAVE CODE
AND #$07 ;STRIP ALL BUT SEG #
LDY SLOTNO
STA SEGMBASE,Y ;NEXT INSTR. RUN FROM NEW SEGMENT
* NEW SEGMENT
  PLA ;GET CODE
  PEA RETURN0 ;where to return to
  AND #$F8 ;STIP OFF SEG# LEAVING SUB #
  LSR
  LSR ;LEAVE SUB# MULTIPLIED BY 2
* GET ADDRESS OF SUB FROM SUBTABL & PUSH ON STACK
  TAY
  LDA SUBTABL0,Y
  PHA
  BRA RESTAXP0 ;RESTORE REGISTERS, RTS TO SUBROUTINE
  MX %11

* RETURN HERE FROM SUBROUTINE

RETURN0 EQU *-1
JSR SAVEAXP0
PLA ;SEG # TO RETURN TO
LDY SLOTNO
STA SEGMBASE,Y ;RETURN TO SEGMENT
BRA RESTAXP0
DS \,$FF ;PUT OBJECT AT NEXT PAGE
On a GS it may be necessary to add a slight delay to \( \Phi 0 \) by buffering the signal.

Drawn by: Chuck Kelly
Programming for the 16 channel I/O card.

Each pin on port A and B can be configured to be either an input or an output. This is done by writing a 1 to the corresponding bit in the data direction register for each pin that is to be an output and writing a 0 for each pin that is to be a input.

The following sample code assumes the I/O card is located in slot #6.

A simple program to read the 8 inputs from port A and echo them to port B.

```
DDRA EQU $C0E3 ; data direction register for port A
DDRB EQU $C0E2 ; data direction register for port B
PORTAEQU $C0E1
PORTBEQU $C0E0

LDA #00
STA DDRA ; make inputs of port A
LDA #$FF
STA DDRB ; make outputs of port B

LOOP LDA PORTA
STA PORTB
BRA LOOP
```

The registers of the 6522 would be located at the following addresses if the card were plugged into slot #6.

<table>
<thead>
<tr>
<th>Address</th>
<th>Write</th>
<th>Read</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C0E0</td>
<td>Output Register B</td>
<td>Input Register B</td>
</tr>
<tr>
<td>$C0E1</td>
<td>Output Register A</td>
<td>Input Register A</td>
</tr>
<tr>
<td>$C0E2</td>
<td>Data Direction Register B</td>
<td></td>
</tr>
<tr>
<td>$C0E3</td>
<td>Data Direction Register A</td>
<td></td>
</tr>
<tr>
<td>$C0E4</td>
<td>T1 Low-Order Latches</td>
<td>T1 Low-Order Counter</td>
</tr>
<tr>
<td>$C0E5</td>
<td>T1 High-Order Counter</td>
<td></td>
</tr>
<tr>
<td>$C0E6</td>
<td>T1 Low-Order Latches</td>
<td></td>
</tr>
<tr>
<td>$C0E7</td>
<td>T1 High-Order Latches</td>
<td></td>
</tr>
<tr>
<td>$C0E8</td>
<td>T2 Low-Order Latches</td>
<td>T2 Low-Order Counter</td>
</tr>
<tr>
<td>$C0E9</td>
<td>T2 High-Order Counter</td>
<td></td>
</tr>
<tr>
<td>$C0EA</td>
<td>Shift Register</td>
<td></td>
</tr>
<tr>
<td>$C0EB</td>
<td>Auxiliary Control Register</td>
<td></td>
</tr>
<tr>
<td>$C0EC</td>
<td>Peripheral Control Register</td>
<td></td>
</tr>
<tr>
<td>$C0ED</td>
<td>Interrupt Flag Register</td>
<td></td>
</tr>
<tr>
<td>$C0EE</td>
<td>Interrupt Enable Register</td>
<td></td>
</tr>
<tr>
<td>$C0EF</td>
<td>Same as Reg 1 Except No &quot;Handshake&quot;</td>
<td></td>
</tr>
</tbody>
</table>

For more information on the 6522 and other hardware I recommend the following books.

Apple II Assembly Language by: Marvin L. De Jong
Howard W. Sams & Co

Microcomputer Electronics by: Daniel L. Metzger
Prentice Hall