PS/2 Auxiliary Device
Data Format and Timing

PS/2 port Interface between Auxiliary device using GMS87C1408 and Host(PC)

DESCRIPTION

This document architects a class of auxiliary pointing devices which includes an extension of the PS/2™ Mouse as specified in the Mouse Technical Reference(IBM™ Pub. No. 68X2229). Typically, this includes such devices as trackballs, scratch pads and joystick. It also includes devices like TrackPoint II which allow the PS/2 Auxiliary Device Port to be used by two or more such devices, either effectively simultaneously or alternately under program control.

Here, the protocol between auxiliary device and host(PC) is only described, that is, data format and timing are shown in the text and figure.

And the PS/2 communication and operation of auxiliary device such as mouse, trackball, scratch pads and joystick is described in next application note.

PS/2 PORT PIN ASSIGNMENT

The auxiliary device is connected to the system by a 6 pin connector. The following shows pin numbers and signals.

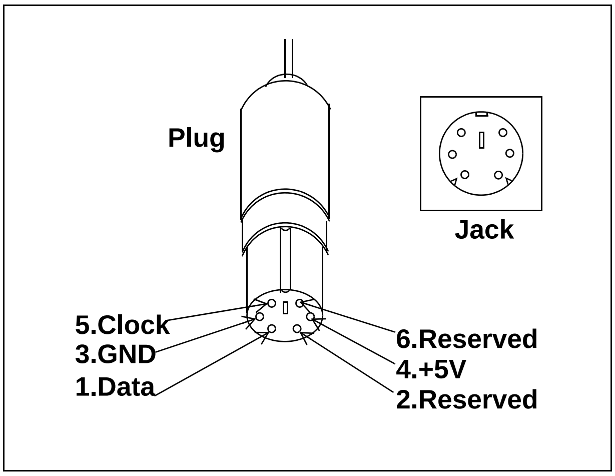


Figure 1. PS/2 Connector

DATA FORMAT

The data transmission between auxiliary device and host(PC) is composed of 11bits. this includes one start bit, eight data bits, one parity bit and one stop bit. The data format is shown in below Figure 2.

Start Bit	Data Bits	Parity Bit	Stop Bit
(1)	(8)	(1)	(1)

Figure 2. Data Format

TIMING CHART AND EXPLANATION

HOST(PC) receiving Data

- ***PC samples data while CLK is low.***

The following describes the typical sequence of events when the Host(PC) is receiving data from the Auxiliary Device

1. The Auxiliary Device checks the CLK line. If the line is inactive(low), output from the Auxiliary Device is not allowed.
2. The Auxiliary Device checks the DATA line. If the line is inactive(low), the Auxiliary Device controller receives data from the Host.
3. The Auxiliary Device checks the CLK line periodically during the transmission at intervals not exceeding 100 microseconds. If the Auxiliary Device finds that the Host is holding the CLK line inactive(low), the byte transmission is terminated. The Host can terminate transmission anytime during the first 10 clock cycles.
4. A final check for terminated transmission is performed at least 5 microseconds after clock ten.
5. The Host can hold the CLK line inactive(low) to inhibit the next transmission.
6. The Host can set the DATA line inactive(low) if it has a byte to transmit to the Auxiliary Device. The DATA line is set inactive(low) when the start bit(always 0) is placed on the DATA line.
7. The Host raises the CLK line to allow the next transmission.

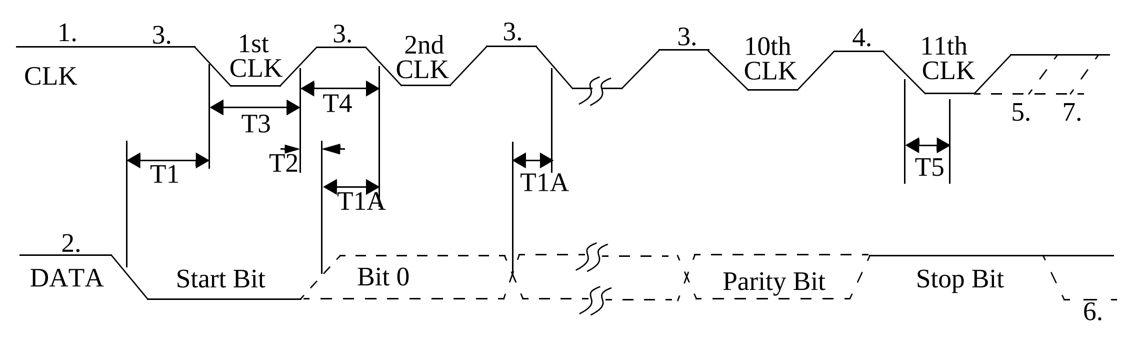


Figure 3. Auxiliary Device Sending Data Timings

Timing	Description	Spec.(Min/Max)
T1	Time from DATA transition to falling edge of CLK1	5/25us
T1A	Time from DATA transition to falling edge of CLK 2-11	5/25us
T2	Time from rising edge of CLK to DATA transition	5/T4-5us
T3	Duration of CLK inactive(low)	30/50us
T4	Duration of CLK active(high)	30/50us
T5	Time to Auxiliary Device inhibit after clock 11 to ensure the Auxiliary Device does not start another transmission	0/50us
TCR	Line contention checking interval	0/100us

Table 1: Auxiliary Device Transmitting Data Timings

Host Sending Data

- ***Auxiliary Device samples data while CLK is high.***

The following describes the typical sequence of events when the Host(PC) is sending data to the Auxiliary Device

1. The Host checks for an Auxiliary Device transmission in process. If a transmission is in process and beyond the 10th clock, the Host must receive the data.
2. The Auxiliary Device checks the CLK line. If the line is inactive(low), an I/O operation is not allowed.
3. The Auxiliary Device checks the DATA line. If the line is inactive(low), the Host has data to transmit.
   The DATA line is set inactive(low) when the start bit(always 0) is placed on the DATA line.
4. The Auxiliary Device sets the CLK line inactive(low). The Host then places the first bit on the DATA line.
   Each time the Auxiliary Device sets the CLK line inactive(after falling edge), the Host places the next bit on the DATA line until all bits are transmitted.
5. The Auxiliary Device samples the DATA line for each bit while the CLK line is active(high).
   Data must be stable within 1microsecond after the rising edge of the CLK line.
6. The Auxiliary Device checks for a positive-level stop bit after the 10th clock. If the DATA line is inactive(low), the Auxiliary Device continues to clock until the DATA line becomes active(high).
   Then the Auxiliary Device clocks the line-control bit and, at the next opportunity, sends a Resend command to the Host.
7. The Auxiliary Device pulls the DATA line inactive(low), producing the line-control bit.
8. The Host can pull the CLK line inactive(low), inhibiting the Auxiliary Device.

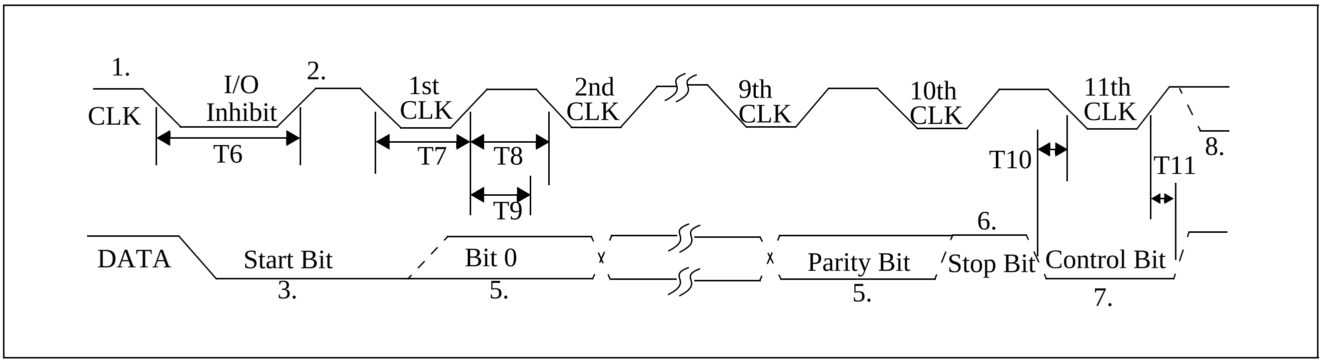


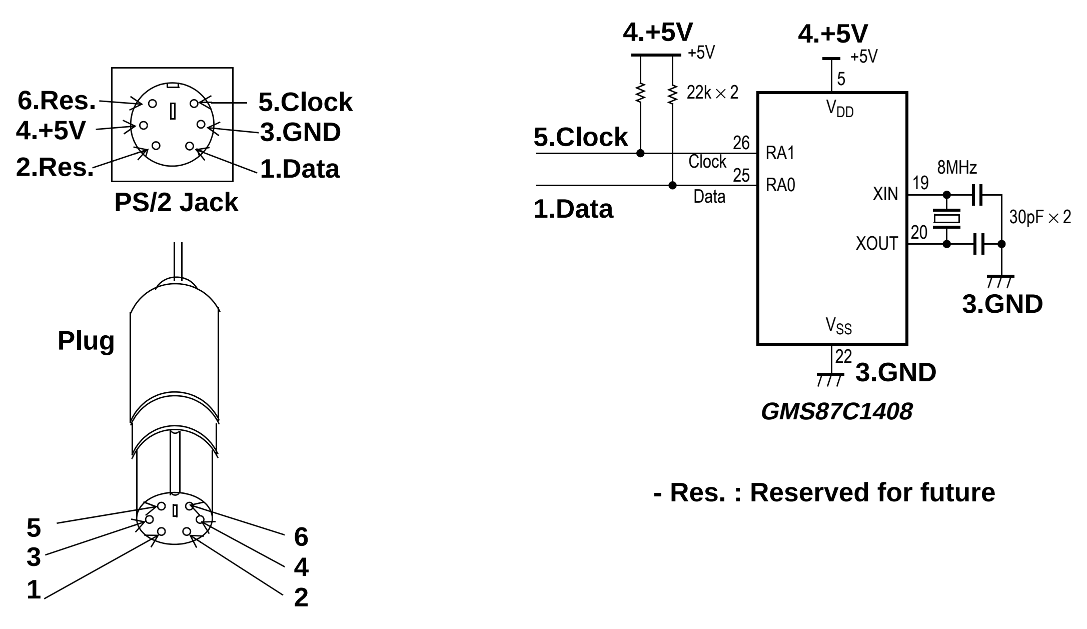
Figure 4. Auxiliary Device Receiving Data Timings (I)

Timing	Description	Spec.(Min/Max)
T6	Auxiliary Device response to Host “request to send”	30us/10ms
T7	Duration of CLK inactive(low)	30us/50us
T8	Duration of CLK active(high)	30us/50us
T9	Time from inactive to active CLK transition, used to time when the Auxiliary Device samples DATA	5/25us
T10	Time from falling edge of line control bit to falling edge of clock 11 CLK	30/50us
T11	Time from rising edge of clock 11 to rising edge of line control bit	0/50us
TCX	Line contention checking interval	0/100us

Table 2: Auxiliary Device Receiving Data Timings (II)

Simple Hardware

A typical hardware connection is shown in Figure 5.



- Res. : Reserved for future

Figure 5. Simple Hardware and Connection

Author: *Sungjae Hwang*
MCU application team

Appendix A:

```
GMS800 series MICOM ASSEMBLER Fri May 11 17:59:03 2001
(PAGE 1)

1                         PAGE    1000
2                ;========================================================================;
3                ;= Copy Right(c) Hynix Semiconductor 2001                               =;
4                ;= All Right Reserved.                                                  =;
5                ;========================================================================;
6                ;= Title: PS/2 Mouse Protocol                                           =;
7                ;= Subject: 1, Transmit data(MSEDATA) to PC through PS/2                =;
8                ;=          2. Get the mouse command from PC throuth PS/2               =;
9                ;=             and Store it to Memory(MSECOMMAND)                       =;
10               ;= Description:                                                         =;
11               ;= Keyboard and mouse are connected to Personal Computer through PS/2   =;
12               ;= Port. The communication through PS/2 port is similar to that of I2C. =;
13               ;= That is, It is use two wires, one is clock and The other is data line.;
14               ;= This program is referred to IBM PS/2 Mouse specification(Ref.II,III) =;
15               ;========================================================================;
16               ;= Device : GMS87C1408                                                  =;
17               ;= OSC    : 8MHz                                                        =;
18               ;= Start Date: 2001, 5,                                                 =;
19               ;= End Date: 2001, 5,                                                   =;
20               ;= S/W    : SP/MCU Application Design Team                              =;
21               ;= Developer: Sungjae Hwang                                             =;
22               ;========================================================================;
23               ;
24               ;Control Registers
25               RA       EQU     0C0H         ;[R/W] RA Port Data Reg.
26               RAIO     EQU     0C1H         ;[W] RA Port Direction Reg.
27               RAFUNC   EQU     0CAH         ;[W] RA Fuction Selection Reg.
28               IRQL     EQU     0E5H         ;[R/W] INT Request Reg. Low
29               WDTIF    EQU     6,0E5H
30               BITIF    EQU     5,0E5H
31               BITR     EQU     0ECH         ;[R] Basic Interval Timer Reg.
32               CKCTLR   EQU     0ECH         ;[W] Clock Control Register
33               WDTR     EQU     0EDH         ;[R],[W] Watchdog Timer Register
34
35
36               ;User Memory
37               MSEDATA  DS      1            ;The data to transmit to PC
38               MSECOMMAND DS      1            ;The command to get from PC
39               COMMANDBUF DS      1            ;Store command from PC.
40               BITCNT   DS      1            ;To count the transmitted bit
41               PARITY   DS      1            ;To check parity
42               ;Constant
43               STACK    EQU     0BFH
44               OUTPUT_MODE EQU     1111_1111B   ;MSESCL=RA.1,MSESDA=RA.0
45               INPUT_MODE EQU     1111_1100B   ;MSESCL=RA.1,MSESDA=RA.0
46               SDA_INPUT_MODE EQU     1111_1110B   ;MSESCL=RA.1,MSESDA=RA.0
47               ;Port Definition
48               MSESDA   EQU     0,RA         ;Mouse Data Definiton
49               MSESCL   EQU     1,RA         ;Mouse Clk Definition
50               ;Macro Definition
51               ble      macro                ;A < #imm jump \1(label)
52                        bcc     \1           ;branch less (after CMP)
53                        endm
54               ;========================================================================;
55               ;= Vector Area                                                          =;
56               ;========================================================================;
57                        ORG     0FFFEH
58 FFFE 00E0              DW      Reset        ; Reset
```

```
              ;=========================================================================;
              ;= Main Routine                                                    =;
              ;=========================================================================;
                       ORG     0E000H       ;GMS87C1408 Start Address
              Reset:
E000 60                di
E001 40                clrg
E002 1EBF              ldx     #STACK       ;Stack Point(00-BFh,OnlyExist 0-page,192byte)
E004 8E                txsp
              ;
E005 1E00              ldx     #0
              Clr0Page:
E007 C400              lda     #0
E009 FB                sta     {X}+
E00A 5EC0              cmpx    #STACK+1     ;0-Page RAM Clear(00-BFh)
E00C 70F9              bne     Clr0Page
              ;
E00E E400C0            ldm     RA,#0000_0000B
E011 E4FCC1            ldm     RAIO,#1111_1100B   ;O RA7~2(Open,Output)
              ;I RA1(PS2 Mouse CLK Input(Initial))
              ;I RA0(PS2 Mouse Data Input(Initial))
E014 E400CA            ldm     RAFUNC,#0    ;Reset Value(RA Port I/O use)
E017 E41EEC            ldm     CKCTLR,#0001_1110B   ;Initial(-001_0111B),BITCLK=1/(8Mhz/512)=64us
E01A E488ED            ldm     WDTR,#1000_1000B   ;WDT(=64us*256*8=131.072ms)
              ;No_useei
              TrnsmitDataToPC:
E01D E4AA00            ldm     MSEDATA,#0AAh   ;Mouse Diagnostics Success Tx Value
E020 3B43E0            call    !XmitMSE1ByteToPC
E023 3B16E1            call    !Delay500us
E026 E40000            ldm     MSEDATA,#00h   ;Mouse ID
E029 3B43E0            call    !XmitMSE1ByteToPC
E02C 3B16E1            call    !delay500us
              MSECommandFromPC:
E02F 33C00A            bbc     MSESCL,MSECommandEnd   ;When MouseCLK(RA.1)=1 & MouseDATA(RA.0)=0,
E032 03C007            bbs     MSESDA,MSECommandEnd   ;The command from PC is exist.
E035 3BB6E0            call    !GetMSE1byteFromPC   ;Stored MouseDataFromPC to MSECOMMAND
E038 C501              lda     MSECOMMAND
E03A E502              sta     COMMANDBUF   ;Execute as Command
              MSECommandEnd:
E03C E488ED            ldm     WDTR,#1000_1000B   ;WDT(=64us*256*8=131.072ms)
E03F B1E5              clr1    BITIF
E041 2FF9              bra     MSECommandEnd
              ;
              ;=========================================================================;
              ;= Sub Routine                                                     =;
              ;=========================================================================;
              ;-------------------------------------------------------------------------;
              ;- Trnsmit Mouse_Data To Personal Computer                         -;
              ;-------------------------------------------------------------------------;
              XmitMSE1ByteToPC:             ;1machine(cpu)cycle:0.25us@8Mhz
E043 C500              lda     MSEDATA
E045 33C0FD            bbc     MSESCL,$     ;R0[R/W],R0DD[W,Write Only]
E048 13C0FD            bbc     MSESDA,$
E04B 21C0              set1    MSESCL
E04D 01C0              set1    MSESDA
E04F E4FFC1            ldm     RAIO,#OUTPUT_MODE   ;[w] Switch output mode
              ;  di                                                              ;MSESCL=RA.1,MSESDA=RA.0
E052 E40803            ldm     BITCNT,#8
E055 E40104            ldm     PARITY,#1    ;odd
E058 11C0              clr1    MSESDA       ;I2C Start Bit("0")
E05A 21C0              set1    MSESCL
              ;*** Time Fix ***
```

```
E05C 3B4AE1            call    !Delay19us   ;76cyc(19us,Total=80cycle(20us))
E05F 31C0              clr1    MSESCL       ;4cyc(1us)
E061 3B22E1            call    !Delay39us   ;156 cycle(39us,Total=160cycle(40us))
E064 21C0              set1    MSESCL       ;4cycle(1us,total=160cycle(40us))
E066 3B52E1            call    !Delay14us   ;56cycle(14us)
              XmitMSE8Bits:
E069 68                ror     A            ;2cyc,Rotate Right through carry(C->b7,b0->C)
E06A EBC010            stc     MSESDA       ;6cyc
E06D 5005              bcc     MSEParityAddZero   ;2(No),4(Yes)
E06F 8904              inc     PARITY       ;4cyc
E071 FF                nop                  ;2cyc
E072 2F04              bra     MSEParityAddOne   ;4cyc
              MSEParityAddZero:
E074 FF                nop                  ;2cyc
E075 FF                nop                  ;2cyc
E076 FF                nop                  ;2cyc
E077 FF                nop                  ;2cyc
              MSEParityAddOne:
E078 21C0              set1    MSESCL       ;4cyc
E07A 3B4AE1            call    !Delay19us   ;76cyc(19us,Total=80cycle(20us))
E07D 31C0              clr1    MSESCL       ;4cyc(1us)
E07F 3B22E1            call    !Delay39us   ;156 cycle(39us,Total=160cycle(40us))
E082 21C0              set1    MSESCL       ;4cycle(1us,total=160cycle(40us))
E084 3B50E1            call    !Delay15us   ;61cyc(15.25us)
E087 AC03DF            dbne    bitcnt,XmitMSE8bits   ;5(No),7(Yes)
              ;---
E08A 6904              ror     PARITY       ;4cyc,to check odd or even(if data is odd then parity is 0)
E08C EBC010            stc     MSESDA       ;6cyc,if C=1(data is odd) then set KBDDATALINE
E08F 21C0              set1    MSESCL       ;4cyc,
E091 3B4AE1            call    !Delay19us   ;76cyc(19us,Total=80cycle(20us))
E094 31C0              clr1    MSESCL       ;4cyc(1us)
E096 3B22E1            call    !Delay39us   ;156 cycle(39us,Total=160cycle(40us))
E099 21C0              set1    MSESCL       ;4cyc,
E09B 3B4CE1            call    !Delay18us   ;72cyc(19us,Total=80cycle(20us))
              ;---
E09E 01C0              set1    MSESDA       ;4cyc, I2C Stop Bit("1")
E0A0 21C0              set1    MSESCL       ;4cyc,
E0A2 3B4AE1            call    !Delay19us   ;76cyc(19us,Total=80cycle(20us))
E0A5 31C0              clr1    MSESCL       ;4cyc,
E0A7 3B22E1            call    !Delay39us   ;156 cycle(39us,Total=160cycle(40us))
E0AA 21C0              set1    MSESCL       ;4cyc
              ;*** Time Fix ***
              ;  ei
E0AC E4FCC1            ldm     RAIO,#INPUT_MODE   ;[w] Switch input mode
              ;MSESCL=RA.1,MSESDA=RA.0
E0AF 3B22E1            call    !Delay39us   ;wait Ack(clk line)
E0B2 33C0FD            bbc     MSESCL,$     ;switched to High after 80us(Low)
E0B5 6F                ret                  ;R0[R/W],R0DD[W,Write Only]
              ;---------------------------------------------------------------------;
              ;- Get PC_Command_For_Mouse(Data) From PC and Store to RCVMSEDATA-;
              ;---------------------------------------------------------------------;
              GetMSE1byteFromPC:
E0B6 33C0FD            bbc     MSESCL,$     ;MouseCLK(RA.1)=1 & MouseDATA(RA.0)=0¿œ¹ß
E0B9 03C0FD            bbs     MSESDA,$
E0BC 21C0              set1    MSESCL
E0BE E4FEC1            ldm     RAIO,#SDA_INPUT_MODE   ;MouseClk=OutputMode
E0C1 E40803            ldm     BITCNT,#8
E0C4 E40104            ldm     PARITY,#1
E0C7 C400              lda     #0
E0C9 E40001            ldm     MSECOMMAND,#0
              ;*** Time Fix
E0CC 31C0              clr1    MSESCL
```

```
E0CE 3B24E1            call    !Delay38us   ;152cyc
E0D1 FF                nop                  ;2cyc
              GetMSE8bits:
E0D2 FF                nop                  ;2cyc
E0D3 21C0              set1    MSESCL       ;4cyc
E0D5 CBC000            ldc     MSESDA       ;4cyc.(Bit0~Bit7)
E0D8 68                ror     A            ;2cyc.
E0D9 3B26E1            call    !Delay37us   ;148cyc
E0DC FF                nop                  ;2cyc.
E0DD 31C0              clr1    MSESCL       ;4cyc
E0DF 3B26E1            call    !Delay37us   ;147cyc.(148cyc)
E0E2 AC03ED            dbne    BITCNT,GetMSE8bits   ;5cyc(No)/7cyc(Yes)
E0E5 E501              sta     MSECOMMAND   ;4cyc.
E0E7 21C0              set1    MSESCL       ;4cyc.Parity Bit(10th Clk)
E0E9 3B22E1            call    !Delay39us   ;156cyc.9th_bit
E0EC 31C0              clr1    MSESCL       ;4cyc.
E0EE 3B22E1            call    !Delay39us   ;156cyc.
E0F1 21C0              set1    MSESCL       ;4cyc.
E0F3 3B4CE1            call    !Delay18us   ;72cyc.,10th_bit
E0F6 11C0              clr1    MSESDA       ;4cyc.Low Setting
E0F8 E4FCC1            ldm     RAIO,#INPUT_MODE   ;5cyc.Change Output(Clk,Data)
E0FB 11C0              clr1    MSESDA       ;4cyc.Low Setting
E0FD 3B4CE1            call    !Delay18us   ;71cyc.(72cyc)
E100 31C0              clr1    MSESCL       ;4cyc.
E102 3B22E1            call    !Delay39us   ;156cyc.
E105 21C0              set1    MSESCL       ;4cyc.
E107 3B4AE1            call    !Delay19us   ;76us
E10A 01C0              set1    MSESDA       ;4us
              ;*** Time Fix
E10C E4FCC1            ldm     RAIO,#INPUT_MODE
E10F 3B4AE1            call    !Delay19us
E112 33C0FD            bbc     MSESCL,$
E115 6F                ret
              ;------------------------------------------------------------------------;
              ;- Time Delay Routine                                             -;
              ;------------------------------------------------------------------------;
              Delay500us:
E116 E41EEC            ldm     CKCTLR,#0001_1110B   ;Bit6(WAKEUP),Bit5(PCWDT),Bit4(WDTON)
              ;Bit3(BTCL),B210(110b=Fxin/512=64us)
E119 FF                nop                  ;64us*256=16.384mS at 8MHZ
E11A FF                nop
              Judge500us:
E11B C5EC              lda     BITR
E11D 4407              cmp     #07          ;64us*7=448us
              ble Judge500us
E11F 50FA   @          bcc     Judge500us
E121 6F                ret
              ;------------------------------------------------------------------------;
              Delay39us:
E122 FF                nop                  ;(39us)
E123 FF                nop
              Delay38us:
E124 FF                nop                  ;(38us)
E125 FF                nop
              Delay37us:
E126 FF                nop                  ;(37us)
E127 FF                nop
E128 FF                nop                  ;(36us)
E129 FF                nop
E12A FF                nop                  ;(35us)
E12B FF                nop
E12C FF                nop                  ;(34us)
```

```
E12D FF                nop
E12E FF                nop                  ;(33us)
E12F FF                nop
E130 FF                nop                  ;(32us)
E131 FF                nop
E132 FF                nop                  ;(31us)
E133 FF                nop
E134 FF                nop                  ;(30us)
E135 FF                nop
E136 FF                nop                  ;(29us)
E137 FF                nop
E138 FF                nop                  ;(28us)
E139 FF                nop
E13A FF                nop                  ;(27us)
E13B FF                nop
E13C FF                nop                  ;(26us)
E13D FF                nop
E13E FF                nop                  ;(25us)
E13F FF                nop
E140 FF                nop                  ;(24us)
E141 FF                nop
E142 FF                nop                  ;(23us)
E143 FF                nop
E144 FF                nop                  ;(22us)
E145 FF                nop
E146 FF                nop                  ;(21us)
E147 FF                nop
E148 FF                nop                  ;(20us)
E149 FF                nop
              Delay19us:
E14A FF                nop                  ;(19us)
E14B FF                nop
              Delay18us:
E14C FF                nop                  ;(18us)
E14D FF                nop
E14E FF                nop                  ;(16us)
E14F FF                nop
              Delay15us:
E150 FF                nop                  ;(15us)
E151 FF                nop
              Delay14us:
E152 FF                nop                  ;(14us)
E153 FF                nop
E154 FF                nop                  ;(13us,call=8cycle)
E155 FF                nop
E156 FF                nop                  ;(12us)
E157 FF                nop
E158 FF                nop                  ;(11us)
E159 FF                nop
E15A FF                nop                  ;(10us)
E15B FF                nop
E15C FF                nop                  ;(9us)
E15D FF                nop
E15E FF                nop                  ;(8us)
E15F FF                nop
E160 FF                nop                  ;(7us)
E161 FF                nop
E162 FF                nop                  ;(6us)
E163 FF                nop
E164 FF                nop                  ;(5us)
E165 FF                nop
E166 FF                nop                  ; 8cyc(4us,call=8cycle)(add+1)
```

```
307 E167 6F                ret                 ; 5cyc
308               ;------------------------------------------------------------------------;
309                        END


        -- 0 Error(s) --

   --- Total Machine Code : 362 Bytes --
```

NOTE: