— COMPUTER TECHNOLOGY —

A Commodore 64-based interface system for the operant laboratory

C. D. L. WYNNE

Universität Konstanz, Konstanz, West Germany

A system of interfaces for the Commodore 64 (C-64) microcomputer for use in the operant laboratory is described. This system frees the experimenter from limitations on the number of I/O lines available on the user port of the C-64 and offers the control of operant chambers for a low cost. Subroutines in machine code to control the interfaces, offering millisecond timing of external events, are presented. A sample BASIC program demonstrates how the subroutines are called from BASIC to run a simple discrimination experiment.

The Commodore 64 (C-64) is an inexpensive and reliable microcomputer that is widely available on both sides of the Atlantic. Aside from its low cost and its availability, the C-64 possesses two additional features that make it especially attractive as a control system in the psychological laboratory. First, three timers are available to the programmer, including one that can be programmed to a resolution of 1 msec; and second, the user port enables simple control of peripheral devices.

The system described here is intended for a laboratory in which each operant chamber is controlled by its own C-64 computer connected with four commercially available interfaces. The combined cost of the computer with disk drive and sufficient interfacing for 24 input and 24 output lines is under \$600 (U.S.). A machine code subroutine described here utilizes the timer chip in the C-64 to record events in the operant chamber with millisecond accuracy; further machine code subroutines utilize the interfaces to control the operation of devices (e.g., reinforcer, stimulus projectors, etc.) in the operant chamber. The machine code subroutines are called as required from programs written in Commodore BASIC; an example of such a program is given that controls a simple concurrent discrimination experiment. The control of other experiments requires only the development of new BASIC programs; the machine code subroutine library presented here does not need to be altered.

SYSTEM COMPONENTS

The Timer

The timer subroutine used here is taken from Hormann and Allen (1987), to which reference should be made for further details. This routine makes use of the two 16-bit registers on CIA chip #2. These are decremented at a rate determined by the system clock (1.0225 MHz for U.S. models; 0.981 MHz for European models; Englisch, 1984). If timer A (TA) is loaded with the value 981 (\$03D5: 1022, \$03FE in the U.S.) and started in recycling mode, then it will flow under 1,000 times per second. Each underflow decrements timer B (TB), and thus the complement of the value in TB gives the number of milliseconds elapsed. Initializing TB with the maximum value of 65535 (\$FFFF) enables a maximum elapsed time of 1.05 min to be measured. If longer elapsed times are required, these may be achieved by replacing the value loaded into TA with a multiple of 981 (or 1022 in the U.S.; see Modifications and Extensions, below).

The DELA Interfaces

The C-64 is equipped with a user port connected primarily with CIA #2; it can be fairly simply programmed but suffers from the drawback of only offering a small number of I/O lines (8 to 12 lines are available; O'Dell & Jackson, 1986). McByte¹ offers a series of interfaces that utilize two of the user port lines to expand very substantially the number of I/O channels available. Three types of cards are available: input cards with 24 TTL input lines (about 50DM [under \$30]); digital output cards with 24 TTL output gates (also about 50DM), and relay output cards with 8 relays capable of switching 120 VAC or 28 VDC (about 100DM [under \$60]). Up to 16 cards of any type can be connected together to one C-64. Due to the size of the user port, not more than 8 lines on one card can be truly simultaneously addressed. This, however, is unlikely to be a practical limitation in the psy-

This paper was prepared while the author was supported by a fellowship from the A. von Humboldt-Stiftung in the laboratories of J. D. Delius. I am most grateful to Juan Delius and to the von Humboldt-Stiftung for their help and support. I also thank Lorenzo von Fersen for helpful comments on an earlier draft. Equipment was provided by grants to J. D. Delius from the Deutsche Forschungsgemeinschaft. Reprints may be obtained from the author at Universität Konstanz, Allgemeine Psychologie, Postfach 5560, D-7750 Konstanz, West Germany.

chological laboratory. If more than three interface cards are in use, the manufacturers recommend that +5 VDC be supplied externally and not from the microcomputer. In practice, the power drain of even one relay card is close to the limits of the 5-V supply on the user port.

Each interface includes a DIP switch by means of which the card address (integer in range 0–15) may be set (we have received interfaces where the DIP switches were right/left reversed). In addition, the input cards and digital output cards each contain 24 lines grouped into three channels of 8 lines each.

Operating the output cards proceeds in four steps. First the card address and channel number (selected by adding 16, 32, or 64 to the card address) must be placed on the user port (data port B of CIA #2; DPB). For the relay cards, the selection of a channel has no effect. Second, this value is sent to the cards by a brief pulse on the serial port line (SPI); this operation illuminates the LED on the selected card. Third, the desired bit pattern for output is set up on the DPB (the output lines are now bitmapped on the DPB). Finally this pattern is sent to the output card by a brief impulse on line 2 of data port A (PA2). Stages 3 and 4 can be repeated at will. Since the default setting of the port is for output, there is no need to reset the cards after output operations.

The procedure for inputting from the cards is slightly more complex. As for output, the card and channel number must first be set up on the DPB and sent to the cards with an impulse to SPI. Then the data direction register B (DDB) and line PA2 must both be set to zero before the card selected will send input to the DPB. A machine code routine then polls DPB and times the delay until an event or events is registered on the selected line or lines (up to a maximum of eight lines on one card and channel). At this point, if the next operation is not input, PA2 should again be set high and DDB reset for output (255, \$FF).

Prior to any operations, the interface cards must be initialized. This involves setting DDB for output, setting PA2 high, and closing the outputs on all cards. The initialization portion of the machine code program presented here also initializes the timer.

Hardware Requirements

Aside from the C-64 computer (with some kind of monitor or television) and DELA interfaces (with external 5-VDC supply) described above, no other hardware is strictly necessary. A disk drive (Commodore Model 1541, 1541-II, or 1571, or a compatible clone model) is highly advisable, however. It should be noted that the interface card only takes TTL voltages (+5 V and 0 V); if non-TTL inputs are required, some additional interfacing will be necessary (a 7400 integrated circuit is a simple possibility; see O'Dell & Jackson, 1986).

SOFTWARE

Overview

The interface and timer software have been written in 6510 machine code to be located at addresses 49152

(\$C000) to 49298 (\$C092). It is, however, fully relocatable, and users with disk drives may prefer to put the program into the cassette buffer (base address 828, \$0332; see Modifications and Extensions, below). The machine code program DELA.MAE, shown in Listing 3, may be loaded into memory with the BASIC loader, DELA.LDR (Listing 2); with an assembler (e.g., MAE from Eastern House Software), or with a monitor (e.g., micromon or supermon).

DELA.BAS

DELA.BAS (Listing 1) is an example of a BASIC program that makes use of the machine code subroutine described below (DELA.MAE), to control four DELA interface cards: three relay cards and one input card. Before DELA.BAS will function, the machine code subroutines must first be loaded (e.g., with DELA.LDR). The experiment involves two response manipulanda (e.g., pecking keys), each of which has associated with it a stimulus presentation device (in-line projector) that can present eight stimuli. On each trial, each pecking key is illuminated with one of eight stimuli. One relay card controls the presentation of stimuli to each pecking key; the third relay card controls the houselight and reinforcing device (e.g., food hopper). Responses to the two keys are registered on the input card. Correct responses are rewarded with 2 sec of access to the food hopper; incorrect responses are punished by a 5-sec blackout.

The first three lines of the program contain, hidden as REMarks, the necessary BASIC statements to test the output cards, as long as the initialization subroutine (SYS 49152; line 60) has been run at least once. Lines 55 and 56 alter the base time unit of the timer machine code to 2 msec, and thus the maximal elapsed time to 2.1 min (see Modifications and Extensions, below). The selection of the input ports to be checked is achieved by POKing values in lines 120 and 130 (these lines are not strictly necessary, since DELA.MAE in any case checks these ports; their inclusion demonstrates how the values in the machine code subprogram can be altered from BASIC). Lines 150 through 170 read from the DATA in lines 2020 through 2060 the stimuli to be presented on each key, and which key is correct. Since all values in lines 2020 and 2040 are powers of two, only one stimulus at a time is presented on each projector. At line 210, the houselight (port 0 on card 3) is turned on before (in lines 230 and 240) stimuli are set on each key. In lines 250 through 300, the program is set to wait for an event on port 0 or 1 of the input card. As soon as an event occurs, lines 310 and 320 record the reaction time and the key pecked. Lines 340 and 350 reset the user port for output, and line 410 darkens the keys before, in line 510, the program determines whether the response should be rewarded or punished, by comparing the number of the pecked key with the relevant value in the array CT. If the response was correct, control passes to the subroutine at line 800, where the food hopper (port 1 of card 3) is activated for 2 sec. Incorrect responses are punished by a 5-sec blackout at lines 520 through 540. In either case, control passes

LISTING 1

DELA.BAS: Sample BASIC Program to Run Concurrent Discrimination Experiment

```

0 REM INPUT "CARD":CARD
1 REM INPUT "MASK":MASK
2 REM GOSUB 1000:GOTO 1
10 REM ** DELA.BAS BASIC PROGRAM FOR TWO KEYS **
20 REM ** EACH WITH 8 STIMULI **
30 REM ** CLIVE WYNNE FEB 1989 **
40 :
50 REM INITIALISE INTERFACES, TIMER
55 POKE 49195,170:REM 2 MSEC
56 POKE 49197,7: REM PRECISION
60 SYS 49152: REM INIT MC PROGRAM
70 NT=16:DIM RT(NT),KY(NT),KO(NT),K1(NT),CT(NT)
80 INPUT"<HOME><DOWN><DOWN><DOWN><DOWN>DATE (DDMM)":DA
90 INPUT" SUBJECT":SU$
100 SU$=SU$+STR$(DA)
110 REM ** SETUP INPUT MASKS **
120 POKE 49269,255-1
130 POKE 49273,255-2
140 REM ** GET STIMULI FOR 2 KEYS **
150 FOR T=1TONT:READ KO(T):NEXT T:REM GET STIM NOS
160 FOR T=1TONT:READ K1(T):NEXT T
170 FOR T=1TONT:READ CT(T):NEXT T:REM GET CORRECT RESPONSE
180 T=1
190 :
200 INPUT"<DOWN><DOWN>PRESS <RETURN> TO START EXPERIMENT":DA
210 CARD=3:MASK=1:GOSUB 1000: REM HOUSELIGHT ON
220 REM ** SET STIMULI ON KEYS **
230 CARD=0:MASK=KO(T):GOSUB 1000
240 CARD=1:MASK=K1(T):GOSUB 1000
250 REM ** WAIT FOR INPUT **
260 POKE 56577,16+2:REM CHANNEL +CARD
270 SYS 49221:REM SEND SP1
280 POKE 56579,0: REM SET USERPORT TO INPUT
290 SYS 49239:REM SET PA2=0
300 SYS 49257:REM TIME
310 RT(T)=PEEK(250)+256*PEEK(251)
320 KY(T)=PEEK(252)
330 REM ** DISABLE TIMER **
340 SYS 49248: REM PA2=1
350 POKE 56579,255: REM USERPORT OUTPUT
360 :
370 :
400 REM ** TURN OFF STIMULI **
410 CARD=0:MASK=0:GOSUB 1000 :CARD=1:GOSUB 1000
420 :
430 :
500 REM ** REWARD OR PUNISH? **
510 IF 254-KY(T)=CT(T) GOTO 800
520 CARD=3:MASK=0:GOSUB 1000:REM OFF HOUSE LIGHT
530 NOW=TI
540 IF TI<NOW+300 GOTO 540
550 REM ** NEXT TRIAL? **
560 T=T+1
570 IF T<=NT GOTO 210
580 REM ***** END OF EXPERIMENT *****
590 SYS 49152:REM RESET ALL PORTS
600 REM ** PRINT OUT DATA **
610 PRINT"<HOME><DOWN><DOWN><DOWN><DOWN>DATA SET ":SU$
620 PRINT"TRIAL STIMO STIM1 KEY CRT RT(S)"
630 FOR T=1 TO NT
640 PRINTTAB(6)KO(T)TAB(12)K1(T) TAB(18)254-KY(T) TAB(23)CT(T)

```

```

TAB(27)RT(T)/500
650 NEXT
660 END
670 :
680 :
800 REM ** REWARD **
810 CARD=3:MASK=3:GOSUB 1000:REM HOPPER UP
820 NOW=TI
830 IF TI<NOW+120 GOTO 830
840 MASK=1:GOSUB 1000: REM HOPPER DOWN
850 GOTO 550
860 :
870 :
1000 REM ** OUTPUT **
1010 REM *****
1030 POKE 56577,CARD:REM SEND CARD ADDRESS
1040 SYS 49221:REM SEND SP1
1050 POKE 56577,MASK
1060 SYS 49232 :REM SEND PA2
1070 RETURN
1080 :
1090 :
2000 REM ** DATA FOR STIMLI NOS **
2010 REM ** KEY 0 **
2020 DATA 1,1,2,2,4,4,8,8,16,16,32,32,64,64,128,128
2030 REM ** KEY 1 **
2040 DATA 1,1,2,2,4,4,8,8,16,16,32,32,64,64,128,128
2050 REM ** CORRECT **
2060 DATA 0,0,1,0,1,0,1,1,0,1,0,1,1,1,0,0

```

to lines 560 and 570, which increment the trial counter and start the next trial, until NT trials have elapsed. At lines 580 ff., the experiment is terminated. All output ports are reset by calling the initialization routine in line 590, and the results are printed out in columns. Note that the raw value of reaction time is divided by 500 to produce an output in seconds.

DELA.LDR

This program, shown in Listing 2, POKEs the necessary machine code instructions in decimal form into the area of memory above address 49152 (\$C000). The DELA.LDR program must be loaded and run before DELA.BAS will function. Alternatively, the BASIC loader program may be incorporated as a subroutine into the desired interface-controlling BASIC program. Only a hardware reset (i.e., turning the power off) will delete the machine code from the computer's memory.

DELA.MAE

Listing 3 gives the source code for the machine language subprogram as written in the MAE assembler. The first section of the program (lines 230 to 380) initializes the DELA interfaces by first setting all the bits of DDB and bit 2 of DDA to output. PA2 is then set high to prepare the cards for output, and, one by one, all the outputs on all cards and channels are closed (set to 0). Lines 400 to 520 complete the initialization routine by initializing the timer as described above. Lines 550 to 600 contain the subroutine to send an impulse on SPI; lines 640

to 670 send an impulse on PA2 by calling the subroutines in lines 700 to 740 and 780 to 820, which set PA2 low and high, respectively. The subroutine to time the delay until a selected event occurs on the selected input card is contained in lines 880 to 1080 (also from Hormann & Allen, 1987).

MODIFICATIONS AND EXTENSIONS

Scanning More Inputs

The machine code program as written scans two ports; as mentioned above, it can easily scan up to eight ports on one channel of one interface card. For each additional port to be scanned, the following pair of machine code lines must be incorporated into DELA.MAE between lines 950 and 960:

```

952 CMP #$FB      ;Is channel 2 closed?
954 BEQ STOPCLOCK ;Yes, stop timer A

```

with different values in line 952 and its equivalents for each port to be checked.

Timer Precision and System Frequencies

The values in lines 420 and 430 of DELA.MAE determine the timer precision by setting the time to underflow of timer A. DELA.MAE is set up for millisecond accuracy; this gives a maximum elapsed time of 1.05 min. Lines 55 and 56 of DELA.BAS show one method of changing the timer: The value 1962 (\$07AA: 2044, \$0804 in the U.S.) is POKEd into the relevant memory loca-

LISTING 2

DELA.LDR: BASIC Program to Load Machine Language Program Shown in Listing 3

```

10 REM ** BASIC PROGRAM TO LOAD MACHINE CODE **
20 REM ** SUBROUTINE FOR DELA INTERFACES      **
30 REM ** CLIVE WYNNE   FEB 1989              **
50 BEGIN=49152:SIZE=146
60 FOR I=BEGIN TO BEGIN+SIZE:READ X:POKE I,X: T=T+X: NEXT
70 IF T<>19046 THEN PRINT "<HOME><DOWN><DOWN><DOWN><DOWN>DATA ERROR: CHECK
DATA STATEMENTS"
80 END
1000 DATA 162,255,142,3,221,173,2
1010 DATA 221,9,4,141,2,221,173,0
1020 DATA 221,9,4,141,0,221,162,240
1030 DATA 142,1,221,32,69,192,169
1040 DATA 0,141,1,221,32,80,192,232
1050 DATA 208,239,169,127,162,213
1060 DATA 160,3,141,13,221,142,4,221
1070 DATA 140,5,221,169,255,141,6
1080 DATA 221,141,7,221,160,81,140
1090 DATA 15,221,96,169,65,141,14
1100 DATA 220,169,1,141,14,220,96
1110 DATA 32,87,192,32,96,192,96,173
1120 DATA 0,221,41,251,141,0,221,96
1130 DATA 173,0,221,9,4,141,0,221
1140 DATA 96,120,162,17,142,14,221
1150 DATA 173,1,221,133,252,201,254
1160 DATA 240,4,201,253,208,243,169
1170 DATA 0,141,14,221,173,6,221,73
1180 DATA 255,133,250,173,7,221,73
1190 DATA 255,133,251,88,184,80,164

```

LISTING 3

DELA.MAE: Source Code in MAE Format of Machine Language Program to Control DELA Interfaces and Time Events

```

0010;Dela machine prog for interfaces
0015;Clive Wynne Feb 1989
0020      .BA $C000 "@0:dela.exe"
0030      .OS
0035;Address Equates
0040CRA1      .DE $DC0E      ;CIA1 control reg A
0050CRA2      .DE $DD0E      ;CIA2 control reg A
0060DDDB      .DE $DD03      ;Data direction reg B
0070DDA      .DE $DD02      ;Data direction reg A
0080DPB      .DE $DD01      ;DATA PORT B (user port)
0090DPA      .DE $DD00      ;DATA PORT A
0100TALD      .DE $DD04      ;Timer low byte
0110TAHI      .DE $DD05      ;Timer high byte
0120CTL0      .DE $DD06      ;Counter low byte
0130CTHI      .DE $DD07      ;Counter high byte
0140ICR      .DE $DD0D      ;Interrupt control register
0150CRB      .DE $DD0F      ;Control register B
0180LATEN      .DE $FA      ;Store latency here
0190KEYVAL      .DE $FC      ;Store port value here
0200;
0210;
0220;Init interfaces
0230      LDX #$FF      ;set data direction reg B
0240      STX DDB      ;to all bits output
0250      LDA DDA      ;set data direction reg A
0260      ORA #$04      ;bit 2 as output (PA2)
0270      STA DDA
0280      LDA DPA      ;set dataport A bit 2 hi
0290      ORA #$04      ;(pa2) hi

```

LISTING 3 (Continued)

```

0300          STA DPA                ;(other DPA bits unchanged)
0310          LDX #$F0              ;select all canals, card 0
0320LOOP     STX DPB                ;set card address on DPB
0330          JSR SP1              ;impulse to sp1
0340          LDA #$00              ;close all outputs
0350          STA DPB              ;by sending $00 to Dataport B
0360          JSR PA2              ;impulse to pa2
0370          INX                  ;next card
0380          BNE LOOP             ;until all cards set
0390;
0400; Initialise CIA2
0410          LDA #$7F              ;mask to defeat CIA IRQ
0420          LDX #$D5              ;timer lo value
0430          LDY #$03              ;timer hi value
0440          STA ICR              ;disable timer from IRQ
0450          STX TALO              ;set timer A to 1 msec
0460          STY TAHI
0470CNTSET   LDA #$FF              ;set up counter
0480          STA CTLO              ;reset counter to FFFF
0490          STA CTHI
0500          LDY #$51              ;link time to count,
0510          STY CRB              ;enable counter
0520          RTS
0530;
0540;
0550;impulse to SP1
0560SP1      LDA #$41              ;shift-register output
0570          STA CRA1              ;i.e. send byte on SP1
0580          LDA #$01              ;shift-register input
0590          STA CRA1              ;i.e. read byte on SP1
0600          RTS
0610;
0620;
0630;
0640;impulse to pa2
0650PA2      JSR PA20              ;set PA2 LO
0660          JSR PA21              ;SET PA2 HI
0670          RTS
0680;
0690;
0700;set pa2 to 0 (set for input)
0710PA20     LDA DPA                ;set dataport A bit 2
0720          AND #$FB              ;lo (PA2 lo)
0730          STA DPA
0740          RTS
0750;
0760;
0770;
0780;set pa2 to 1 (for output)
0790PA21     lda DPA                ;set dataport A
0800          ora #$04              ;hi (PA2 hi)
0810          sta DPA
0820          rts
0830;
0840;
0850; modified millsec timer for dela interface
0860;
0870;
0880; time latency to respond
0890          SEI                  ;disable interrupts
0900          LDX #$11              ;set & start timer a
0910          STX CRA2
0920POLL      LDA DPB                ;check user port B

```

LISTING 3 (Continued)

```

0930      STA KEYVAL      ;store in zero page
0940      CMP #$FE      ;is channel 0 closed?
0950      BEQ STOPCLOCK ;Yes, stop timer A
0960      CMP #$FD      ;No, is channel 1 closed?
0970      BNE POLL      ;No, keep polling
0980STOPCLOCK LDA #$00  ;Stop the clock
0990      STA CRA2      ;(set 0 in control reg A)
1000      LDA CTLO      ;get counter lo value
1010      EOR #$FF      ;complement for latency
1020      STA LATEN      ;& store it
1030      LDA CTHI      ;same for counter hi byte
1040      EOR #$FF
1050      STA LATEN+1
1060      CLI           ;return irqs to system
1070      CLV           ;branch always to cntset
1080      BVC CNTSET
1090      .en
//

```

tions as low byte, high byte, respectively. This value is double the European clock frequency; it thus doubles the time till timer underflow to 2.1 min but reduces the precision of the timer to 2 msec. Alternatively, the values in lines 420 and 430 of DELA.MAE could be replaced; or the last value (i.e., 213) in line 1050 and second value (i.e., 3) in line 1060 of DELA.LDR could be altered (in this case, the checksum value in line 70 would also have to be changed). The maximum duration that can be measured in this way is 69 min, with a precision of 66 msec when timer A is loaded with 64746 (\$FCEA; for U.S. models, 64-msec precision when TA is loaded with 65408; \$FF80).

Hormann and Allen (1987) report that their timer program is highly accurate. In practice, the accuracy of DELA.BAS is limited by the five lines of BASIC (lines 250 through 300) required to start the timer. This small constant error could be removed by recasting these lines in machine code as shown in Listing 4.

Relocation

DELA.MAE is fully relocatable; all that is required is to change the base address in line 20 of DELA.MAE, or the value of BEGIN in line 60 of DELA.LDR. Then, of course, all calls in DELA.BAS to the machine language subroutine must be altered. If, for example, the program is relocated to the cassette buffer (base address 828), it is necessary to subtract 48324 (49152 - 828) from all the

values following SYS calls in DELA.BAS and from the POKE addresses in lines 55, 56 and 120, 130.

Further Possibilities

Clearly the concurrent discrimination experiment described here represents only a small part of the capabilities of a system that can be fitted with up to 384 I/O lines. The use of hardware interrupts (Wright & Dawson, 1988) would simplify the programming of more elaborate experiments. It is not necessary to restrict experiments using the C-64 with DELA interface system to those whose data can easily be analyzed on the C-64; user port interfaces available from a variety of sources adapt the user port to a RS232 serial interface and enable the transfer of data from the C-64 to a computer more suited to intensive data analysis.

REFERENCES

- ENGLISH, L. (1984). *Das Maschinensprache Buch für Fortgeschrittene zum Commodore 64* [The machine code book for experts on the Commodore 64]. Düsseldorf: Data Becker.
- HORMANN, C. A., & ALLEN, J. D. (1987). An accurate millisecond timer for the Commodore 64 or 128. *Behavior Research Methods, Instruments, & Computers*, **19**, 36-41.
- O'DELL, J. W., & JACKSON, D. E. (1986). The Commodore 64 and an interface system for controlling operant chambers. *Behavior Research Methods, Instruments, & Computers*, **18**, 339-341.
- WRIGHT, R. D., & DAWSON, M. R. W. (1988). Using hardware interrupts for timing visual displays and reaction-time key interfacing on the Commodore 64. *Behavior Research Methods, Instruments, & Computers*, **20**, 41-48.

NOTE

1. The DELA interfaces, Digital Input Card (Model 1005003), Digital Output Card (Model 1005004), and Relay Output Card (Model 1005002) are available from McByte GmbH, Maastrichter Str. 23, D-5000 Cologne 1, West Germany.

LISTING 4

Modification of DELA.MAE to Start Timer from Machine Code

```

LDA #$12 ;channel and card address
STA DPB  ;on user port
JSR SP1  ;Impulse on SP1
LDA #$00 ;set user port
STA DDB  ;for input
JSR PA20 ;set PA2 low
JSR $C069 ;start timer

```