8051 – Timers and Serial Port

EE4380 Fall 2001

Class 10

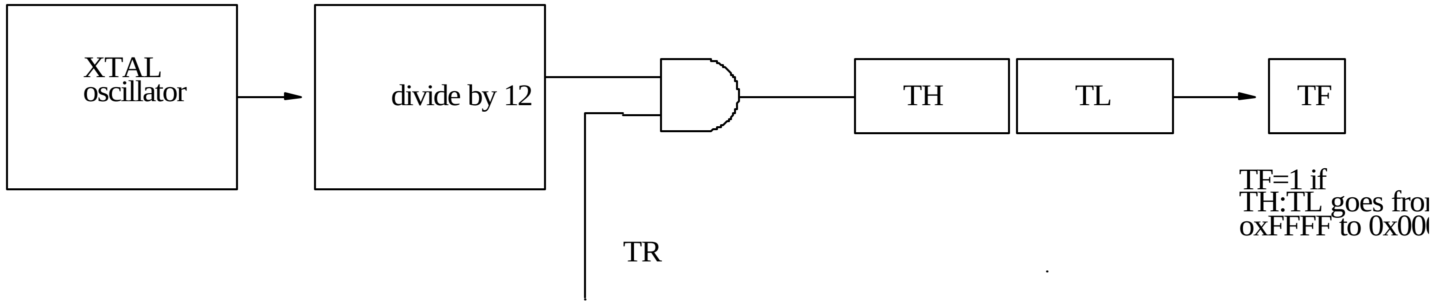
Pari vallal Kannan

Center for Integrated Circuits and Systems

University of Texas at Dallas

Timer: Mode –1 Operation (recap)

- 16 bit counter.
  - Load the counter with a number and set TR, to start counting
  - When the counter rolls over to 0x0000, it sets the TF flag and raises the TF interrupt if enabled



Timer for Time Measurement

- Timer can be used to do measure elapsed time
  - Useful for scheduling routine tasks
  - Similar to "cron" functionality
  - Not as accurate as an RTC, but cheap !
- Timer's clock is 1/12 of the 8051 clock
  - 8051 clock of 11.0592MHz ➔Timer clock of 921.6KHz
  - Time period for one "count" is 1/921.6K = 1.085us
  - Time spent for a count sequence to "roll-over" is
    - Number of counts x 1.085us
- Ex: Timer loaded with 0xFFF2
  - Number of counts to rollover to 0x0000 is 0xFFFF-0xFFF2 +1 = 14
  - Time elapsed = 14 x 1.085us

Time measurement (contd.)

- How to calculate the initial load values for a given time delay requirement T ?
  - Divide T by 1.085us to get n
  - Find m = 65536 – n
  - Convert m to hex, m = 0xUUVV
  - Load TH ← 0xUU and TL ← 0xVV
- For larger delays ?
  - Repeat inside a loop
  - Introduce a number of additional instructions (nop), before enabling the timer again
  - Go for an RTC

Timer : Other modes

- Mode 0
  - Exactly like Mode1, but it is a 13bit timer
  - Count sequence is from 0x0000 to 0x1FFF
- Mode 2
  - 8 bit timer, with auto reload
  - Load the count value in TH and enable the timer
  - 8051 loads TL with TH (TL ← TH)
  - When TL rolls-over to 0x00, timer raises TF flag (and interrupt)
  - After TF flag is cleared by ISR / code, TL is automatically reloaded with TH again and the cycle repeats

Timers as Counters

- Counters are devices how many times a particular event has occurred
  - How many 1's in a bit stream ?
  - How many widgets passed the sensor in an assembly line ?
  - How many dogs walked past my doggie door ?
- Counters increment their count when they receive a signal (count pulse)
- 8051 timers can serve as counters
  - C/T bit in TMOD reg has to be 1 for counter operation
  - Two external pins on 8051 to give the count pulses
    - P3.4 (T0, pin 14) : external count pulse for Timer0
    - P3.5 (T1, pin 15) : external count pulse for Timer1

Counter Example

- Count the pulses on pin T1 (P3.5) and display the counter value on P2. Counter is in mode 2

```
START:    mov TMOD, #01100000B              ;counter 1, mode 2, C/T=1
          mov TH1, #0             ;count from 0x00 to 0xFF
          setb P3.5               ;configure P3.5 as input
AGAIN:    setb TR1                ;enable counter
BACK:     mov A, TL1              ;read TL1 value
          mov P2, A               ;display it on P2
          jnb TF1, back           ;poll for TF1, could use INT1 also
          clr TR1                 ;stop counter
          clr TF1                 ;clear TF1 flag
          sjmp AGAIN              ;while(1)
```

Timers : External Gate

- External gate provides the facility of controlling the timer with an external device
  - Push buttons maybe used to enable/disable timer
  - Snooze button in an 8051 based clock !
- Set GATE=1 in TMOD, then the timer can be controlled from an external pin
  - Pin P3.2 (INT0) for Timer0
  - Pin P3.3 (INT1) for Timer1
- With GATE=1, Timer is enabled iff
  - TRx is set by software (setb TR0)
  - AND, INT0 (Pin P3.2) has to be pulled HIGH by hardware

Serial Communication

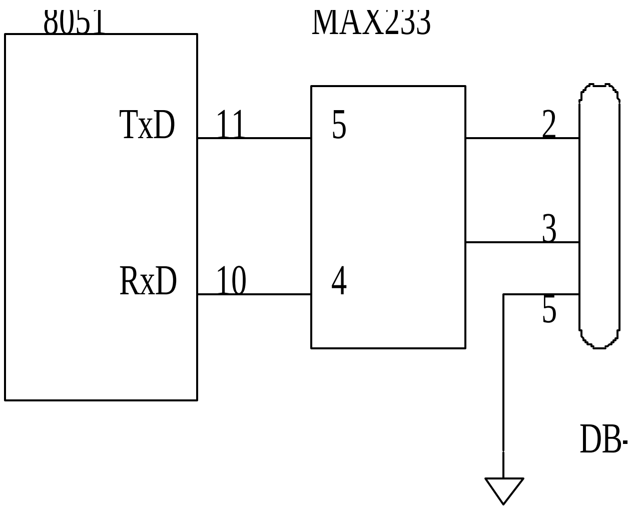
- Serial Vs Parallel Transfer of data
- Simplex, Duplex and half-Duplex modes
- Synchronous, Asynchronous, UART, USART
- Framing
  - Start bit, Stop bit, mark, space
  - Start bit, LSB, MSB, Stop bit
  - Optional parity bit
  - Stop bit can be one or two bits
- Data transfer rate
  - Bps, baud
- RS232 protocol
  - Non TTL compatible logic levels (-3 to –25 for 1 and +3 to +25 for 0)

RS232 Pins

- Too many signals, but most are unused in a microprocessor system
- In non-handshaking mode, only three signals
  - Pin2 : RxD – received data
  - Pin3 : TxD – Transmitted data
  - Pin5 : GND
- For 8051 to PC serial port (COMx) connection, use null-modem connection
  - RxD of 8051 system to TxD of PC
  - TxD of 8051 system to RxD of PC
  - GND to GND
  - Need to set transfer mode to use software flow control (XON/XOF)

RS232 Line driver

- RS232 uses TLL-incompatible logic levels
- Need Line drivers to interface 8051 to Rs232 protocol
- MAX232, MAX233 most commonly used line drivers
  - Dual channels
  - Single supply, +5V operation
  - MAX233 needs no external capacitors



8051 Serial Port

- 8051 has an internal UART
  - Baud rate is set by Timer1
- Control Registers
  - SBUF : Serial Buffer Register
    - Data moved to SBUF is Tx-ed serially
    - Serial data Rx-ed is stored by 8051 in SBUF
  - SCON : Serial Control Register
    - Program the mode (start bit, stop bit, data bits length)
      - Only Mode 1 (8, 1, 1) is of interest, as others are obsolete
    - Receive enable/disable
    - RI and TI, receive and transmit interrupts

Setting the Baud rate

- Timer 1is the timing controller for serial port in 8051
- Clock for the Timer1 in the UART is
  - XTAL /12 /32 = 28,800Hz (for XTAL = 11.0592MHz)
  - Set SMOD (bit 7 of PCON reg) to program 8051 to use 1/16 multiplier
    - XTAL / 12 / 16 = 56,700Hz
    - Effectively doubles the baud rate
- Timer1 has to be programmed in
  - Mode 2, 8bit auto reload mode
  - Load TH1 with the required value
- TH values
  - Baud Rate: 9600 = 28800/3 ➔ TH1 = -3 = 0xFD
  - Baud Rate: 2400 = 28800/12 ➔ TH1 = -12 = 0xF4

SCON Register

- SCON.0 = RI
  - Receive interrupt flag. Valid byte in received in SBUF
- SCON.1 = TI
  - Transmit interrupt flag. Byte in SBUF was completely transmitted.
- SCON.4 = REN
  - Receive enable. Set to enable reception. Clr for transmit only
- SCON.7:SCON.6 = SM0:SM1
  - Serial mode setting
  - 01 = Mode 1 is the widely used mode
    - 8bit data, 1start bit and 1 stop bit
- All other bits to be set to 0

Examples: Transmit a character

- Transfer ASCII “A” serially at 9600 baud continuously

```
START:  mov TMOD, #20H     ;T1 is mode2
        mov TH1, #-3       ;9600 baud
        mov SCON, #50H     ;8b, 1stop, 1start, REN enabled
        setb TR1           ;start T1
AGAIN:  mov SBUF, #’A’     ;letter A is transmitted
HERE:   jnb TI, HERE       ;poll TI until all the bits are transmitted
        clr TI             ;clear TI for the next character
        sjmp AGAIN         ;while(1)
```

Example: Receive Data

- Receive bytes serially and display them on P1, continuously.

```
START:  mov TMOD, #20H      ;T1 in mode 2
        mov TH1, #-3        ;9600 baud
        mov SCON, #50H      ;8b, 1start, 1stop
        setb TR1            ;start T1
HERE:   jnb RI, HERE        ;wait until one byte is Rx-ed
        mov A, SBUF         ;read the received byte from SBUF
        mov P1, A           ;display on P1
        clr RI              ;ready to Rx next byte
        sjmp HERE           ;while (1)
```

Serial Ports with Interrupts

- Using serial port with interrupts is THE way it was intended to be used.
- Both the RI and TI flags raise the Serial interrupt (S0), if it is enabled.
- Simple Case
  - Transmit is polling based (Poll TI flag) and Receive is interrupt driven
  - Transmit is interrupt driven and Receive is polling based (Poll RI flag)
- In these cases, the ISR of S0 will check for the appropriate flag and either copy data to or from SBUF

Serial Ports with Interrupts

- General Case
  - 8051 is in full duplex mode, I.e. receives and transmits data continuously
  - Both Transmit and Receive is interrupt driven
- Write the ISR for S0 such that
  - ISR must first check which one of RI and TI raised the S0 interrupt
  - If RI is set, then read data from SBUF to a safe place and clear RI
  - If TI is set, then copy the next character to be transmitted onto SBUF and clear TI.

Example : Simple case

- 8051 gets data from P1 and sends it to P2 continuously while receiving from Serial port. Serial port data is to be displayed on P0

```
        org 0
        ljmp MAIN                  ;avoid the IVT
        org 23H                    ;serial port ISR
        ljmp SERIAL
        org 30H
MAIN:   mov P1, #0FFH          ;P1 as input port
        mov TMOD, #20          ;T1 in mode 2
        mov TH1, #-3           ;9600 baud
        mov SCON, #50H         ; 8b, 1start, 1stop
        mov IE, #10010000B ;enable S0 interrupt
        setb TR1                   ;enable T1
BACK:   mov A, P1
        mov P2, A
        sjmp BACK
```

```
        org 100H
SERIAL: jb TI, TRANS
        mov A, SBUF   ;copy received data
        mov P0, A     ;display in on P0
        clr RI        ;clear RI
        RETI
TRANS:  clr TI     ;do nothing
        RETI        ;ISR does not handle TX
        end
```