CpE 390: Microprocessor Systems

Lecture 9

68HC12 Serial Interface – SCI

Why Serial Communication?

- Parallel data transfer requires many I/O pins.

- Many I/O devices do not have high data rate to justify the use of parallel data transfer.

- Data synchronization for parallel transfer is difficult to achieve over a long distance.

- Cost consideration

What is SCI?

- A interface designed to transfer data only in asynchronous mode that utilizes the EIA-232 standard.

What is SPI?

- A protocol proposed by Motorola to facilitate the data exchange between the microcontrollers designed by Motorola and peripheral devices.

Asynchronous Serial Data Communication

- Often used for data communication between a DTE (data termination equipment) and a DCE (data communication equipment) with or without a modem.

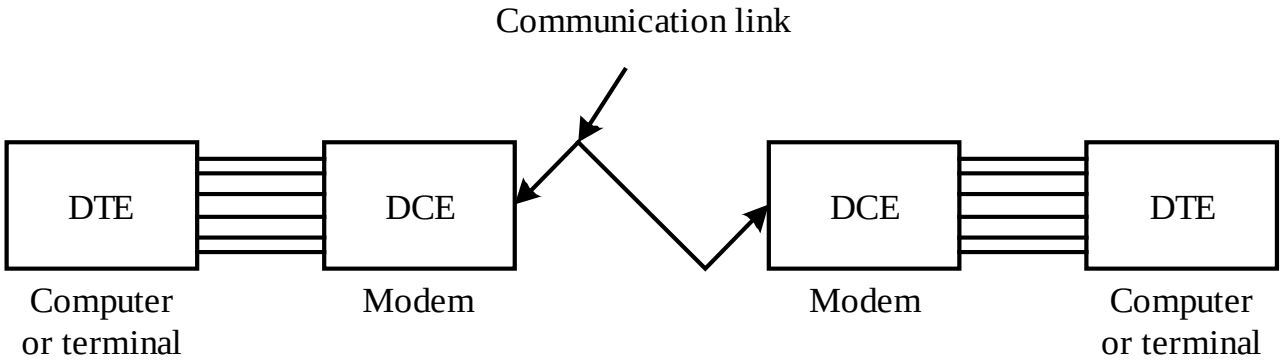


Figure 9.0 A data communication system

- there are two kinds of data communication links:
  1. simplex link: only communicate in one direction
  2. half-duplex link: two-direction link, only one direction at a time
  3. full-duplex link: communicate in both directions simultaneously

Types of communication link configuration

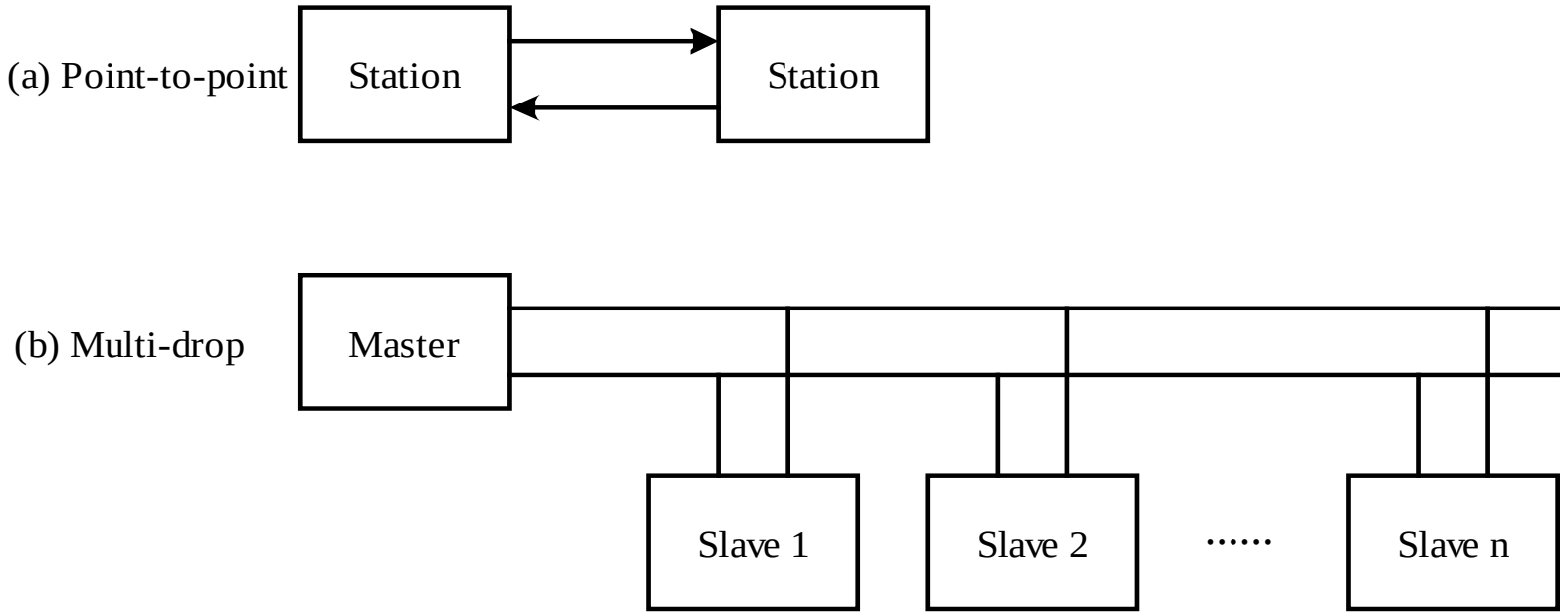


Figure 9P.2 Point-to-point and multi-drop communication links

The RS232 Standard

-was the most widely used physical level interface for data communication
-specifies 25 interchange circuits for DTE/DCE use
-was established in 1960 by Electronics Industry Association (EIA)

EIA-232-E Mechanical Specification

- Specifies a 25-pin connector
- Specifies exact dimensions of each pin.

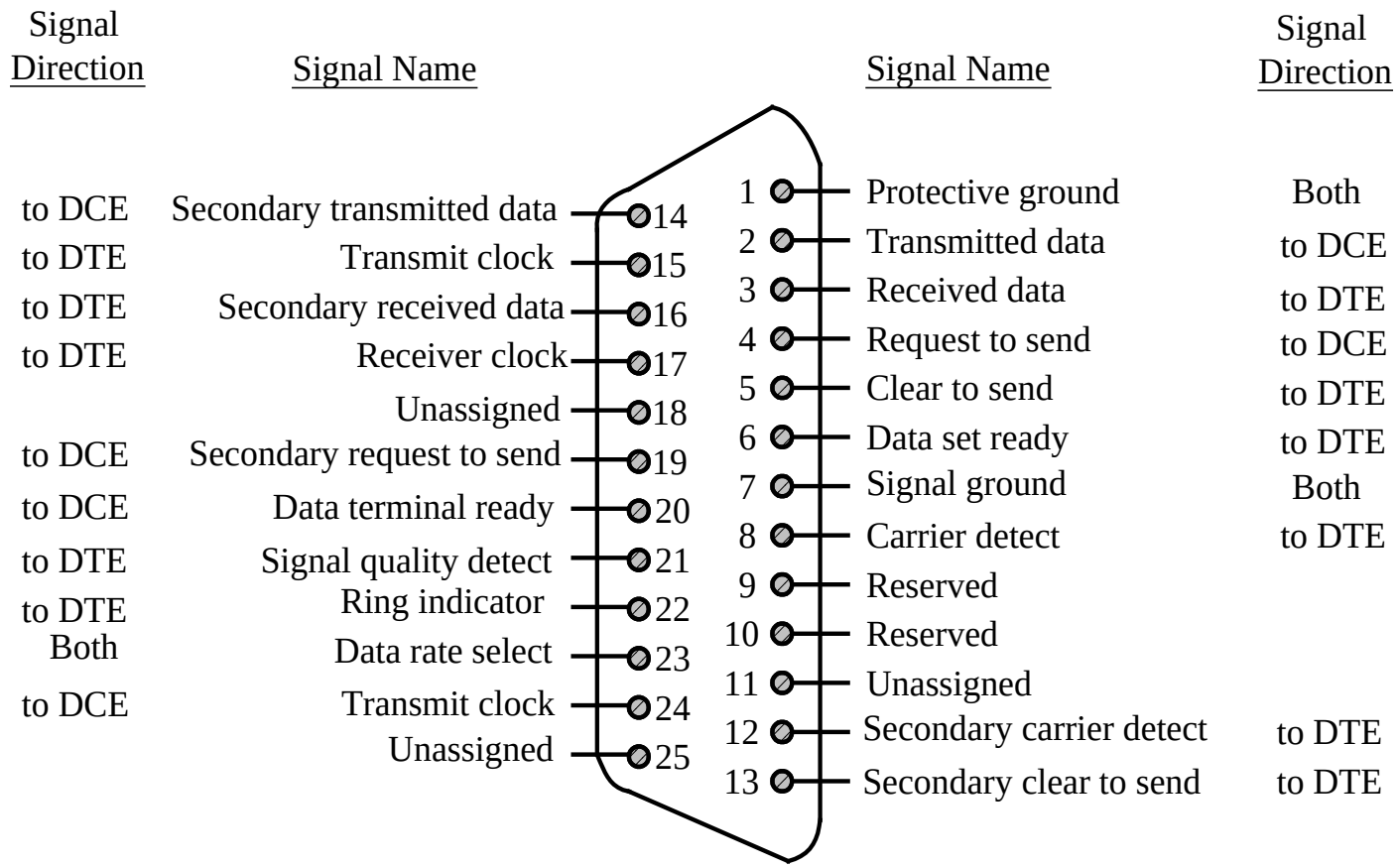


Figure 9.1 EIA-232-E connector and pin assignment

EIA-232-E Procedural Specification

- Define the sequence of events that occurs during data transmission

Case 1. Two DTEs connected via a point-to-point link using a modem.

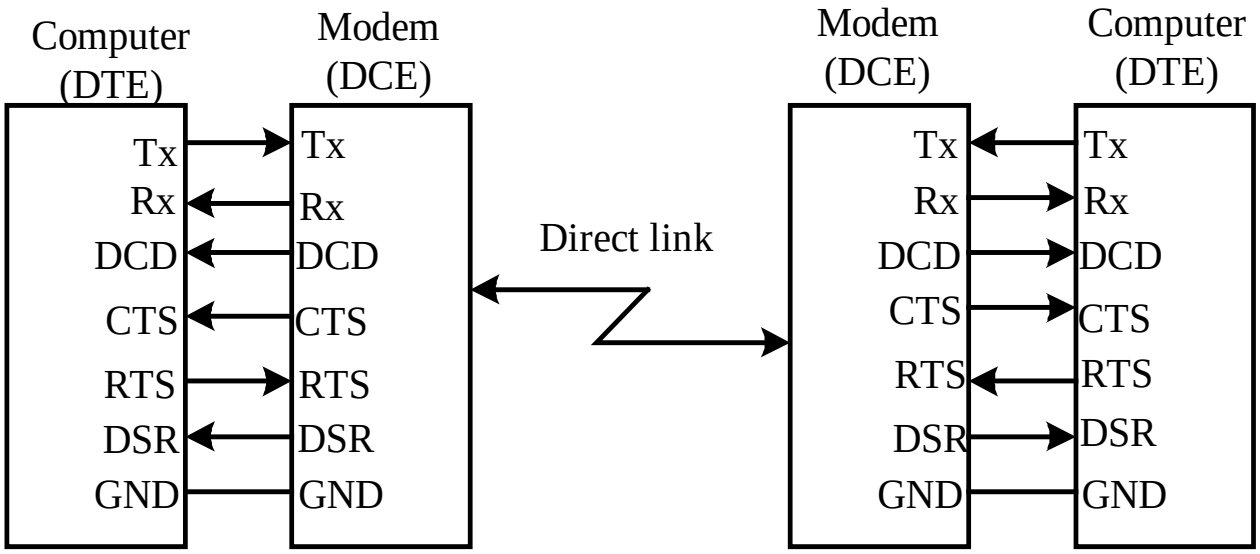
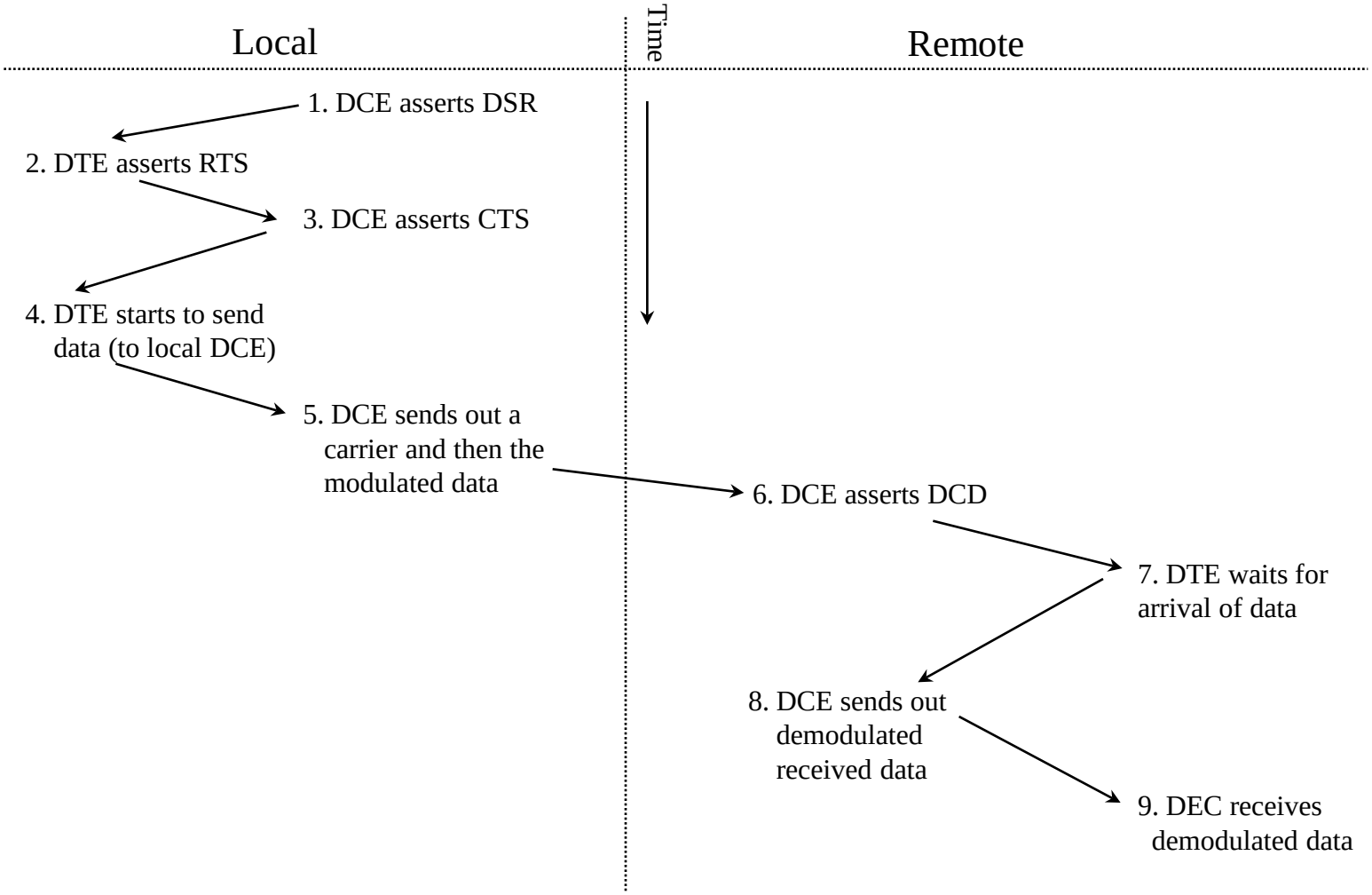


Figure 9.2 Point-to-point asynchronous connection

Sequence of events occurred during data transmission over dedicated link



Case 2. *Two DTEs exchange data through a public phone line*

- The signal DTR is used by the DTE to indicate its intention to make a call or accept a call.
- The signal RING is used by the DCE to indicate that there is an incoming call.

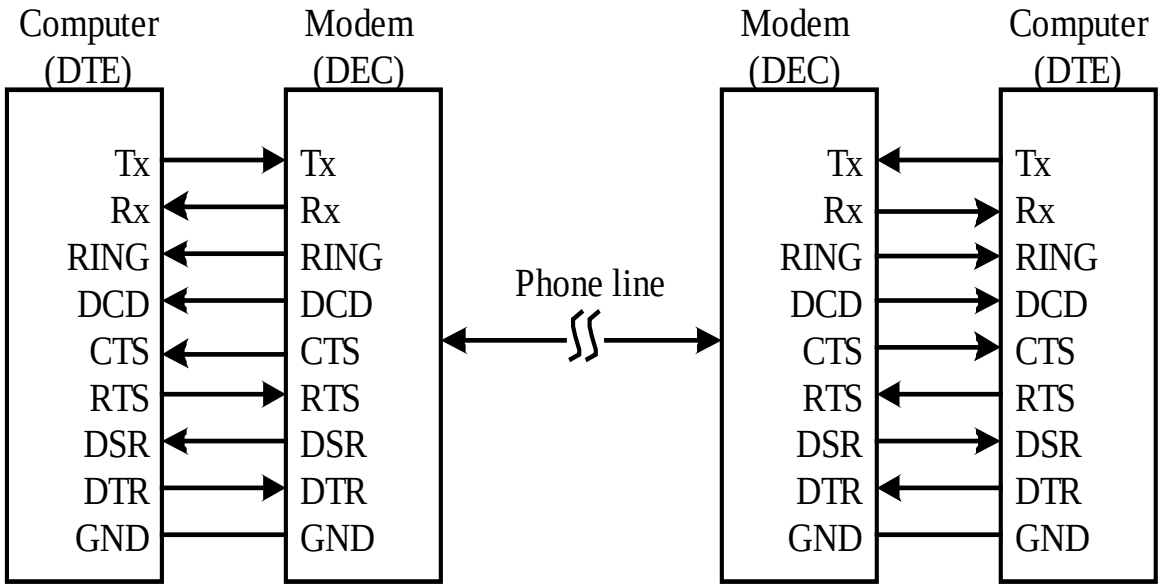
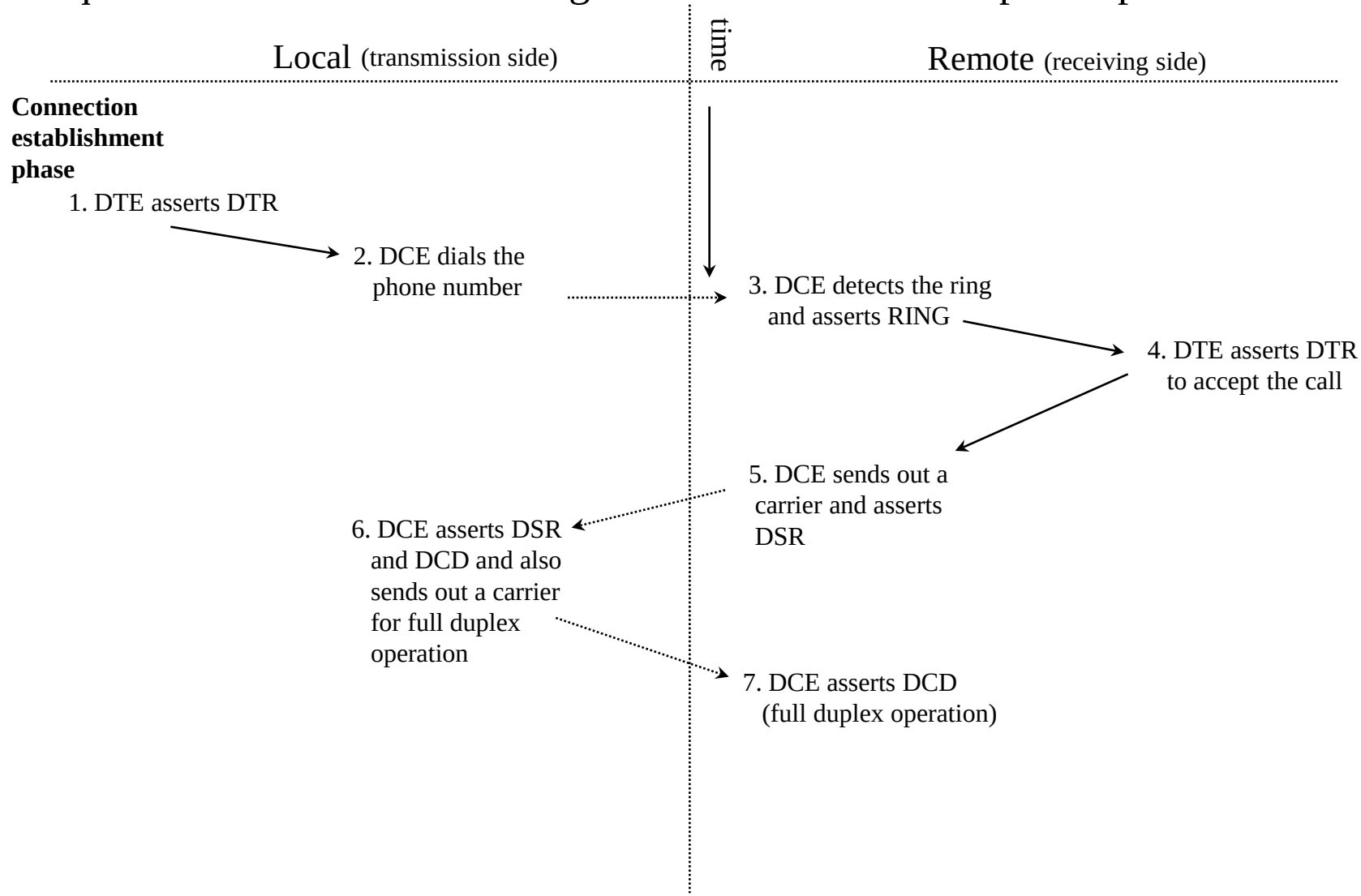
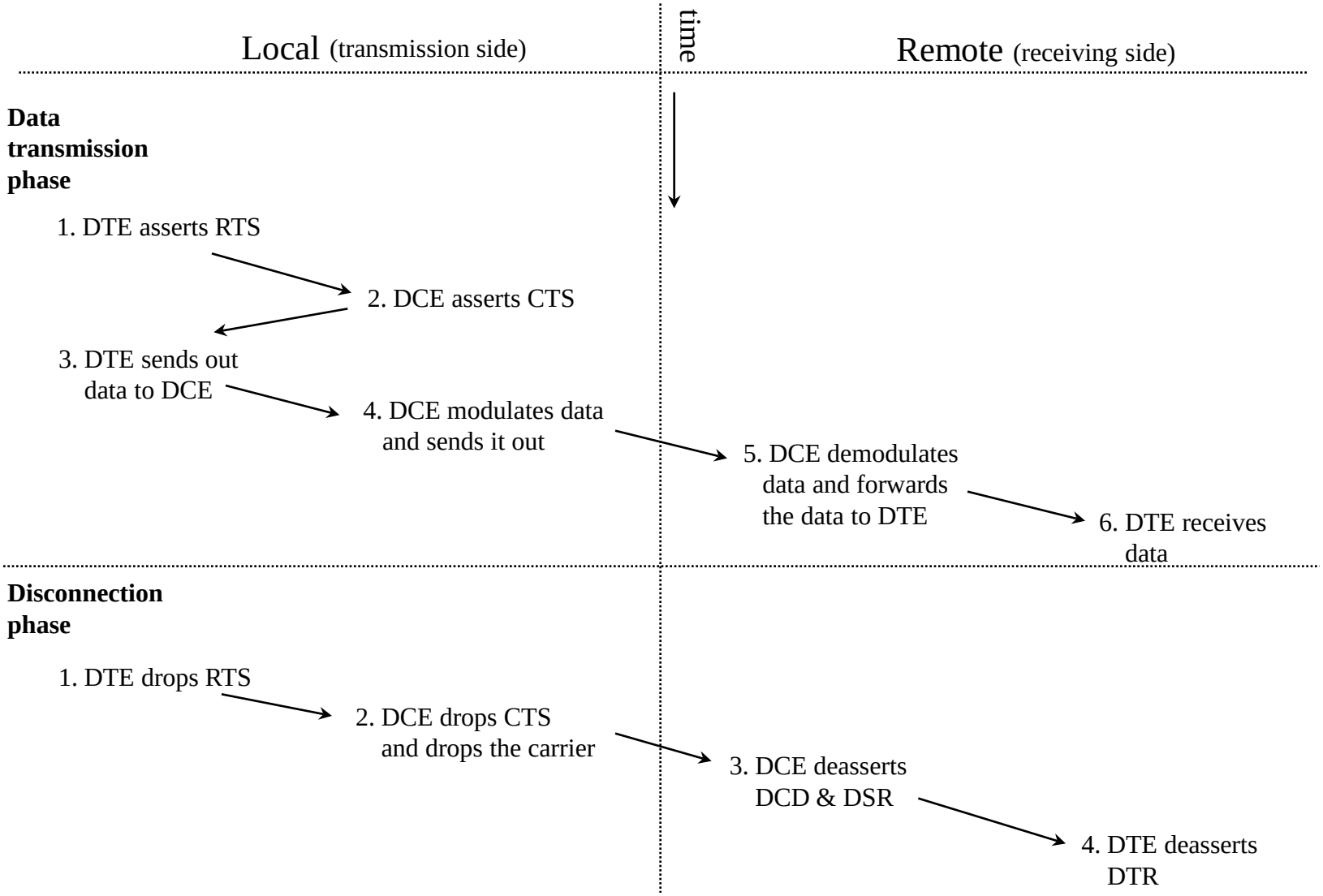


Figure 9.3 Asynchronous connection over public phone line

Sequence of events occur during data transmission over public phone line



Sequence of events occur during data transmission (continued)



Data format for asynchronous data communication

- data is transmitted character by character bit-serially
- a character consists of

1. one start bit (0),
2. 8 data bits,
3. an optional parity bit,
4. one, or one and a half, or two stop bits (1)
5. least significant bit is transmitted first
6. most significant bit is transmitted last

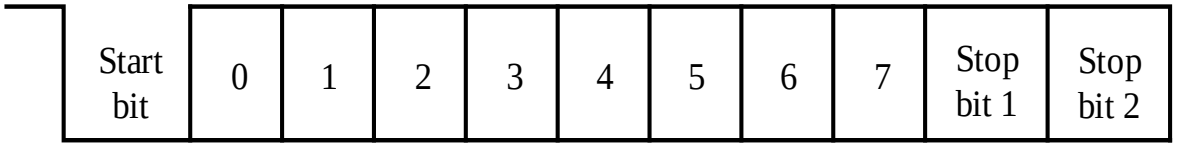


Figure 9.4 The format of a character

How to Detect the Arrival of Start Bit?

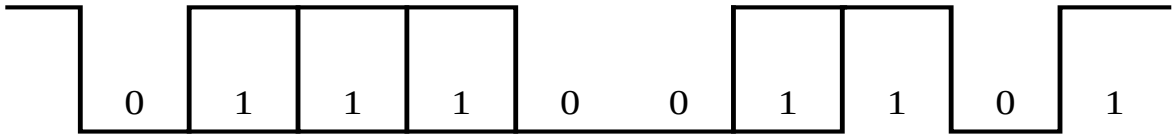
- Use a clock signal with frequency at least 16 times that of the data rate to sample the RxD signal.
- When the RxD pin is idle (high) for at least three sampling times and a falling edge follows, the SCI circuit checks the third, fifth, and seventh samples after the first sample. If the majority of them are low, then the start bit is considered detected.

How to Determine the Logic Value of a Data Bit?

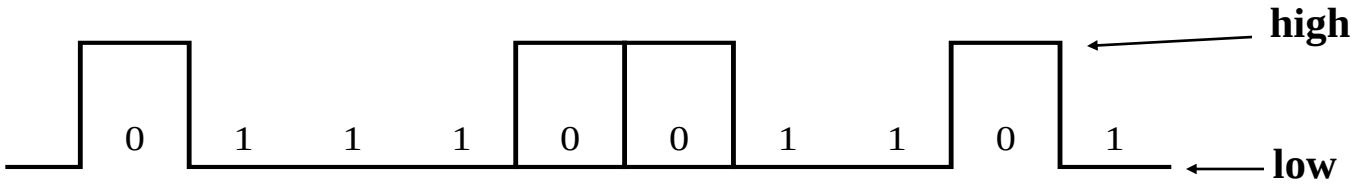
- Use a clock signal with frequency at least 16 times that of the data rate to sample the incoming data.
- Take the majority function of the eighth, ninth, and tenth samples. If the majority of them are 1s, then the logic value is determined to be 1.

Example 9.1 Sketch the output of the letter **g** when it is transmitted using the format of one start bit, 8 data bit, and 1 stop bit.

Solution:
The ASCII code of letter **g** is $67 or %01100111. This code will be followed by a stop bit. The output from the DTE should be:



(a) output waveform on microcontroller interface



(b) output waveform on EIA-232-E interface

Figure 9.6 Data format for letter **g**

Data transmission errors

- *Framing error*: a character is not properly framed by a stop bit
- *Receiver overrun*: one or more characters received but not read by the CPU (overlap)
- *Parity error*: odd number of bits change value

Null Modem connection

- Connect the PC to a single-board computer (demo board), both are DTEs. EIA standard does not allow for a direct connection between two DTEs. A null modem is needed.

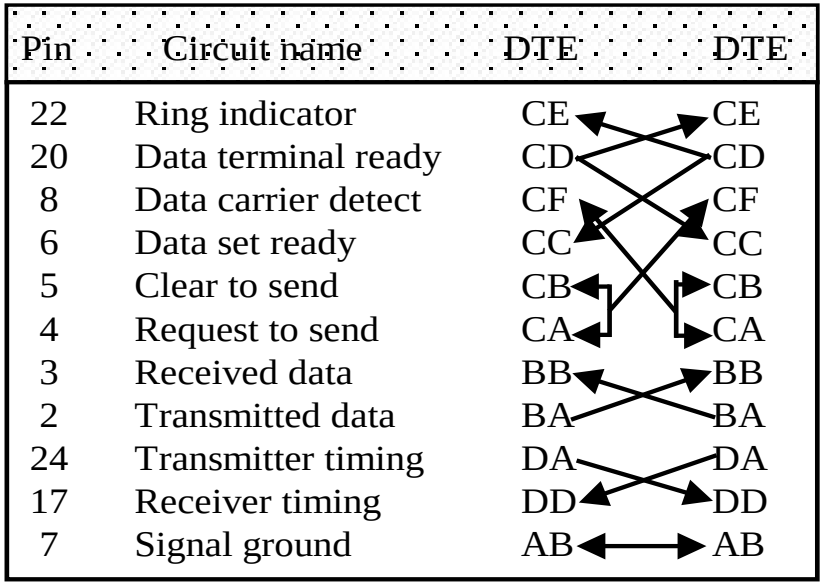


Figure 9.7 Null Modem connection

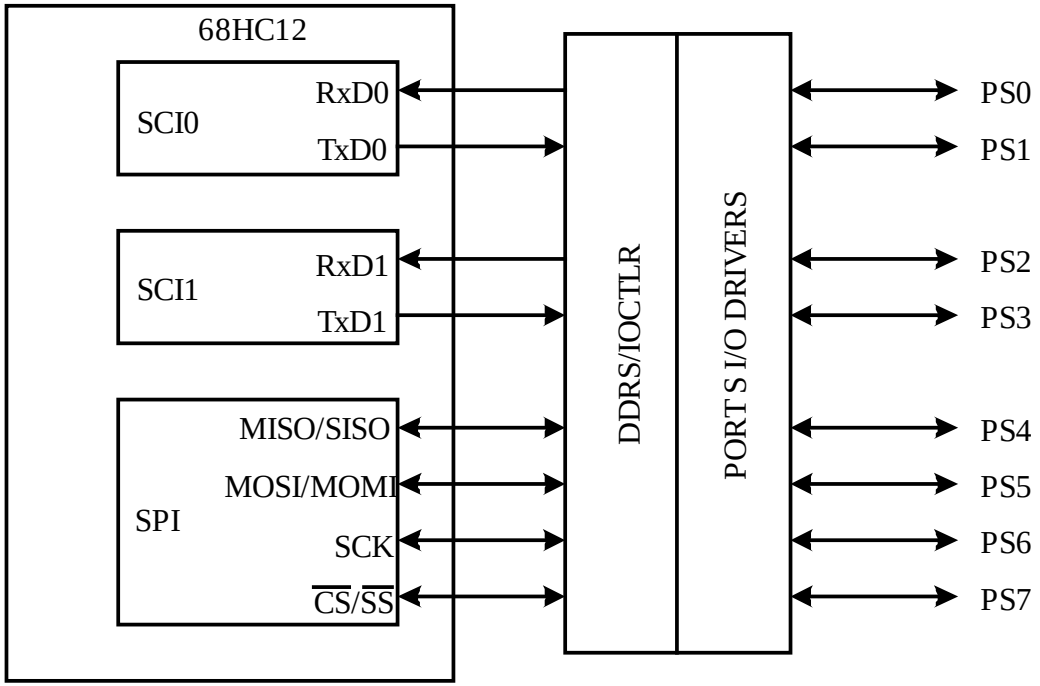
The 68HC12 SCI Subsystem

- Use frame as the data format, consisting of one start, eight or nine data bits, and one stop bit.

-One SCI channel uses two signal pins - PS0 (and PS2) pin for RxD, and PS1 (and PS3) pin for TxD.

- The SCI has the capability to send break to attract the attention of other party of communication.

- A **break** is defined as the transmission or reception of logic 0 for a frame or longer time.



Note. B family members (912B32, 912BC32, and 912BE32) do not have SCI1

Figure 9.8 Multiple serial interface (SCIs and SPI) block diagram

Baud Rate Generation

- Use a 13-bit counter to generate this clock signal. This circuit is called ***baud rate generator***.

- This baud rate generator divides down the P clock to derive the clock signal for reception and transmission.

Table 9.2 Baud rate generation

Desired SCI baud rate	Baud rate divisor for P = 4.0 MHz	Baud rate divisor for P = 8.0 MHz
110	2273	4545
300	833	1667
600	417	833
1200	208	417
2400	104	208
4800	52	104
9600	26	52
14,400	17	35
19,200	13	26
38,400	--	13

Registers associated with SCI

- The 16-bit baud rate control register (SCxBD) control the generation of the baud rate.
- The baud rate is determined by the following formula:

  SCI baud rate = $f_P \div (16 \times BR)$

	7	6	5	4	3	2	1	0	
	BTST	BSPL	BRLD	SBR12	SBR11	SBR10	SBR9	SBR8	$00C0
value after reset	0	0	0	0	0	0	0	0	

(a) SCI baud rate control register high (SC0BDH/SC1BDH)

	7	6	5	4	3	2	1	0	
	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0	$00C1
value after reset	0	0	0	0	0	1	0	0	

(b) SCI baud rate control register low (SC0BDL/SC1BDL)

BTST -- reserved for test function
BSPL -- reserved for test function
BRLD -- reserved for test function

Figure 9.9 SCI baud rate control register

Registers associated with SCI

-Each SCI channel has two control registers (SCxCR1 & SCxCR2). x = 0 or 1

- *Character format, parity checking, wakeup method are programmed via the SCxCR1.*

- *Transmit and receive interrupts enabling, transmission and reception enabling, wakeup enabling and break sending are programmed via the SCxCR2 register.*

- Each SCI channel has two status registers (SCxSR1 and SCxSR2).

- *Data transmission and reception status, idle line status, and reception errors are recorded in the SCxSR1 register.*

- *The SCxSR2 has only one bit common to all 68HC12 members: bit 0. Bit 0 indicates whether there is an reception in progress.*

- Each SCI channel has a 16-bit data register. When 8 data bit format is used, these 8 bits are stored in the lower byte.

	7	6	5	4	3	2	1	0	
	LOOPS	WOMS	RSRC	M	WAKE	ILT	PE	PT	$00C2/$00CA
value after reset	0	0	0	0	0	0	0	0	

LOOPS -- SCI loop mode/single-wire mode enable bit
- 0 = SCI transmit and receive sections operate normally
- 1 = SCI receive section is disconnected from the RxD pin and the RxD pin is available as general-purpose I/O.

WOMS -- wired-or mode for serial pins
- 0 = pins operate in a normal mode with both high and low drive capability.
- 1 = each pin operates in an open drain fashion if that pin is declared as an output.

RSRC -- receiver source bit

When LOOPS = 1, the RSRC bit determines the internal feedback path for the receiver.
- 0 = receiver input connected to the transmitter internally (not TxD pin).
- 1 = receiver input connected to the TxD pin

M -- mode bit (select character format)
- 0 = one start, eight data, one stop bit
- 1 = one start, nine data, one stop bit

WAKE -- wakeup by address mark/idle bit
- 0 = wakeup by idle line
- 1 = wakeup by address mark (last data bit set)

ILT -- idle line type bit
- 0 = short idle line mode enabled
- 1 = long idle line mode detected

PE -- parity enable bit
- 0 = parity disabled
- 1 = parity enabled

PT -- parity type bit (for both transmit and receive)
- 0 = even parity selected
- 1 = odd parity selected

Figure 9.10 SCI control register 1 (SC0CR1/SC1CR1)

Table 9.3 Loop mode functions

LOOPS	RSRC	DDRS1	WOMS	Function of Port S bit 1/3
0	x	x	x	Normal operation
1	0	0	0/1	Loop mode without TxD output (= high impedance)
1	0	1	0	Loop mode with TxD output (CMOS)
1	0	1	1	Loop mode with TxD output (open drain)
1	1	0	x	Single-wire mode without TxD output (pin is used as receiver input only, TxD = high impedance)
1	1	1	0	Single-wire mode without TxD output (pin is used as receiver input only, TxD = high impedance)
1	1	1	1	Single-wire mode for the receiving and transmitting (open drain)

	7	6	5	4	3	2	1	0	
	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK	$00C3/$00CB
value after reset	0	0	0	0	0	0	0	0	

TIE -- transmit interrupt enable bit
- 0 = TDRE interrupt disabled
- 1 = SCI interrupt requested when TDRE status flag is set.

TCIE -- transmit complete interrupt enable bit
- 0 = TC interrupt disabled
- 1 = SCI interrupt requested when TC flag is set

RIE -- receiver interrupt enable bit
- 0 = RDRF and OR interrupts disabled
- 1 = SCI interrupt requested when RDRF status flag or OR status flag is set

ILIE -- idle line interrupt enable bit
- 0 = IDLE interrupt disabled
- 1 = SCI interrupt requested when IDLE status flag is set

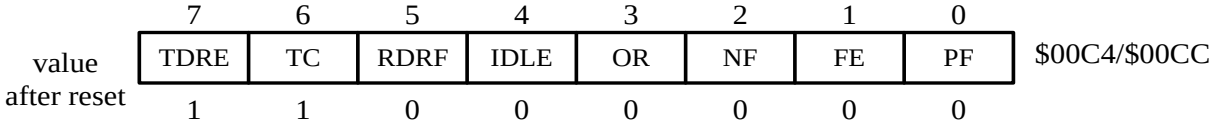
TE -- transmit enable bit
- 0 = transmitter disabled
- 1 = SCI transmit logic is enabled and the TxD pin is dedicated to the transmitter. The TE bit can be used to queue an idle preamble

RE -- receiver enable
- 0 = receiver disabled
- 1 = enable the SCI receive circuitry

RWU -- receiver wakeup control bit
- 0 = normal SCI receiver
- 1 = enables the wakeup function and inhibits further receiver interrupts. Normally, hardware wakes the receiver by automatically clearing this bit.

SBK -- send break bit
- 0 = break generator off
- 1 = generate a break code, at least 10 or 11 contiguous 0s. As long as SBK remains set, the transmitter sends 0s.

Figure 9.11 SCI control register 2 (SC0CR2/SC1CR2)



TDRE -- transmit data register empty flag
- 0 = SC0DR (SC1DR) is busy
- 1 = Any byte in the transmit data register is transferred to the serial shift register so new data may now be written to the transmit data register.

TC -- transmit complete flag
- 0 = transmitter busy
- 1 = transmitter idle

RDRF -- receiver data register full flag
- 0 = SC0DR empty
- 1 = SC0DR full

IDLE -- idle line detected flag
- 0 = RxD line active
- 1 = RxD line idle

OR -- overrun error flag
- 0 = no overrun
- 1 = overrun detected

NF -- noise error flag

Set during the same cycle as the RDRF bit but not set in the case of an overrun (OR)
- 0 = unanimous decision
- 1 = noise on a valid start bit, any of the data bits, or on the stop bit

FE -- framing error flag

Set when a 0 is detected where a stop bit was expected. Clear the FE flag by reading SC0SR1 (SC1SR1) with FE set and then reading SC0DR (SC1DR).
- 0 = stop bit detected
- 1 = zero detected rather than a stop bit

PF -- parity error flag
- 0 = parity correct
- 1 = incorrect parity detected

Figure 9.12 SCI status register 1 (SC0SR1/SC1SR1)

	7	6	5	4	3	2	1	0	
	SCSWAI	0	0	0	0	BRK13	TXDIR	RAF	$00C5/$00CD
value after reset	0	0	0	0	0	0	0	0	

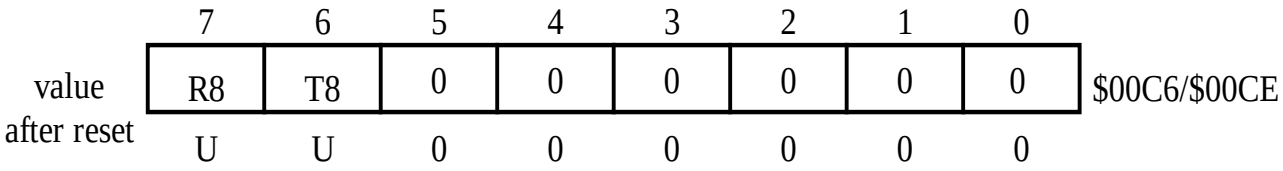
SCSWAI: serial communications stop in wait mode (available in 912DG128 only)
0 = SCI clock operates normally
1 = Halt SCI clock generation when in WAIT mode

BRK13: break transmit character length (available in 912D60 only)
This bit determines whether the transmit break character is 10 or 11 bit respectively 13 or 14 bit long.
0 = break character is 13 or 14 bit long
1 = break character is 10 or 11 bit long

TXDIR: transmitter pin data direction in single-wire mode (available in 912D60 only)
This bit determines whether the TxD pin is going to be used as an input or output, in the single-wire mode of operation. This bit is relevant only in the single-wire mode of operation.
0 = TxD pin to be used as an input in single-wire mode
1 = TxD pin to be used as an output in single-wire mode

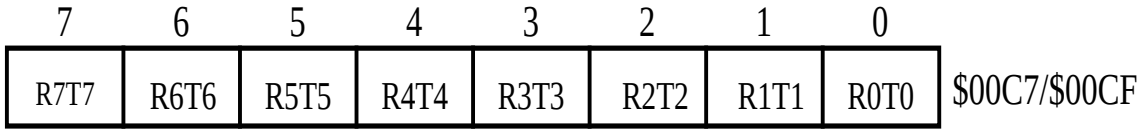
RAF: receiver active flag
RAF is set when the receiver detects a logic 0 during the RT1 time period of the start bit search. RAF is cleared when the receiver detects an idle character.
0 = no reception in progress
1 = reception in progress

Figure 9.13 SCI status register 2 (SC0SR2/SC1SR2)



Bit 5 to 0 cannot be written but will be read as 0s

Figure 9.14 SCI data register high (SC0DRH/SC1DRH)



Unaffected by reset

Figure 9.15 SCI data register low (SC0DRL/SC1DRL)

Example 9.2 Write an instruction sequence to configure the SCI channel 0 to operate with the following parameters:

- 9600 baud (P clock is 8 MHz)
- one start bit, 8 data bits, one stop bit
- no interrupt
- address mark wakeup
- disable wakeup initially
- long idle line mode
- enable transmit and receive
- no loop back
- disable parity checking

Solution: The following instruction sequence will configure the SCI0 properly:

```
movb   #$00,SC0BDH   ; set up baud rate
movb   #52,SC0BDL    ;      “
movb   #$0C,SC0CR1
movb   #$0C,SC0CR2
```

Interfacing SCI with EIA-232-E

- The SCI uses 0 V and 5 V to represent 0 and 1
- The EIA-232 signal Tx cannot be driven by the SCI TxD signal without translation
- The EIA-232 signal Rx cannot drive the SCI RxD signal without translation
- Voltage level translation is required for the SCI signals to drive and be driven by the EIA-232 signals since EIA-232 signals are opposite to SCI signals.
- Examples of EIA-232 driver chips include:
  (1) LT1080/1081 from Linear technology
  (2) ST232 from SGS Thompson
  (3) ICL232 from Intersil
  (4) MAX232 from MAXIM
  (5) DS14C232 from National Semiconductor
  These chips are pin-compatible.
- The DS14C232 from National Semiconductor will be used in the following illustration.

Interfacing the 68HC12 SCI to the EIA-232 using the DS14C232 chip (a voltage translation circuit) and implements the NULL modem connection so that this connection can talk to a PC directly.

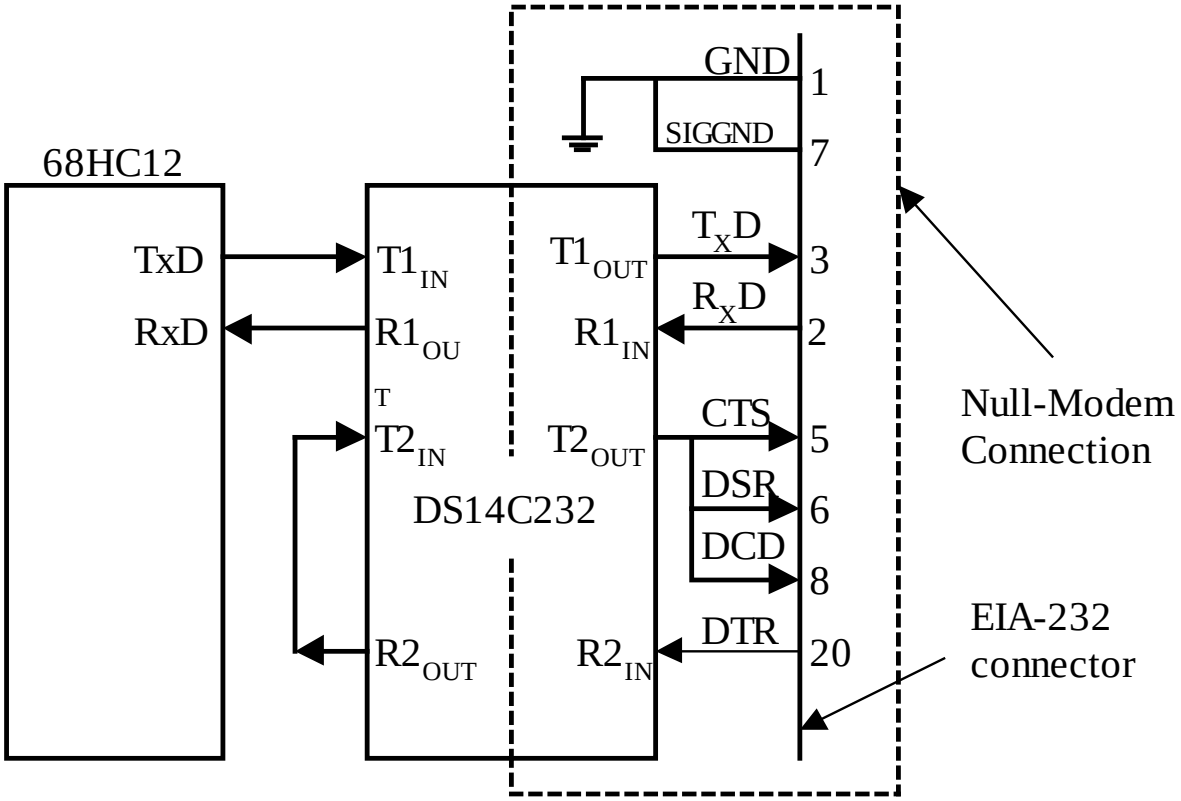


Figure 9.17 Diagram of the SCI and EIA-232 circuit connection

Example 9.3 Write a subroutine to send a break to the communication port controlled by the SCI0 interface. The duration of the break is approximately 40,000 E clock cycles, or 5 ms at 8 MHz.

Solution:

- A break character is represented by ten or eleven consecutive zeros and can be sent out by setting the bit 0 of the SC0CR2 register.

```
#include "d:\miniide\hc12.inc"

sendbrk   bset    SC0CR2,$01     ; turn on send break
          jsr     wait_5ms
          bclr    SC0CR2,$01     ; turn off send break
          rts
wait_5ms  movb    #$90,TSCR      ; enable TCNT and fast flag clear
          movb    #$00,TMSK2     ; set prescale factor to 1
          movb    #$01,TIOS      ; select OC0 function
          ldd     TCNT
          addd    #40000
          std     TC0            ; start an OC0 operation
          brclr   TFLG1,$01,*    ; wait for 5 ms
          rts
```

Example 9.4 Write a subroutine to output the character in accumulator A to the SCI0 channel using the polling method.

Solution:

- The subroutine will wait until the bit 7 of SC0SR1 register is set before sending out the character in accumulator A.

```
#include "d:\miniide\hc12.inc"
putc_sc0        brclr SC0SR1,$80,*   ; wait for TDRE to be set
                staa  SC0DRL         ; output the character
                rts
```

Example 9.5 Write a subroutine to read a character from the SCI channel 0 using the polling method. The character will be returned in accumulator A.

Solution: Assembly function is as follows:

```
#include "d:\miniide\hc12.inc"

getc_sc0  brclr   SC0SR1,$20,*      ; wait until RDRF bit is set
          ldaa    SC0DRL        ; read the character
          rts
```

Example 9.7 Write a subroutine that inputs a string from the SCI channel 0. The string is terminated by the carriage return character and must be stored in a buffer pointed to by index register X.

Solution:

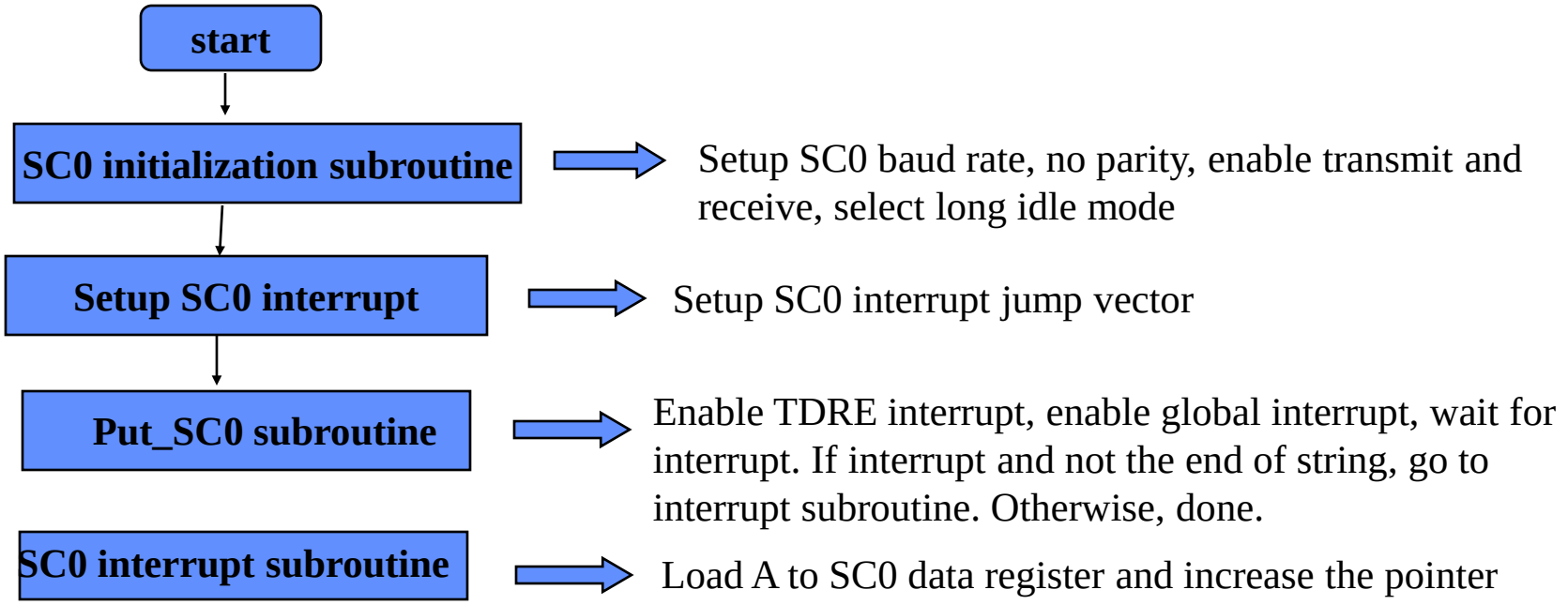
- *This subroutine will call getc_sc0 repeatedly until the carriage return character is received.*

```
gets_sc0   jsr   getc_sc0
           cmpa #CR       ; is the character a carriage return?
           beq  exit
           staa 1,x+      ; save the character in the buffer pointed to by X
           bra  gets_sc0  ; continue
exit       clr  0,x       ; terminate the string with a NULL character
           rts
```

Example 9.8 Use interrupt-driven approach to write a subroutine to output a string pointed to by index register X. The program is to be executed on a board that contains the D-Bug12 monitor. Write an instruction sequence to test your subroutine.

Solution:

- *This subroutine will enable SCI interrupt and stay in a loop to wait for transmit register empty interrupts.*
- *The SCI interrupt service routine output the character in A and return.*



The subroutine is as follows:

```
puts_sc0  bset SC0CR2,$80     ; enable TDRE interrupt
          cli                 ; enable global interrupt
loop      ldaa        0,x     ; wait for TDRE interrupt to occur
          bne  loop           ;
          bclr  SC0CR2,$80    ; reach the end of string, disable TDRE interrupt
          sei                 ; disable global interrupt
          rts

sc0_isr   staa  SC0DRL        ; store string to SC0 data register
          inx                 ; increase the pointer
          stx  3,sp           ; update the copy of X in the stack (transfer x via stack)
          rti
```

The instruction sequence for testing the previous subroutine:

```
#include "d:\miniide\hc12.inc"
sc0_vec_no    equ     11
setuservector equ     $F69A
CR            equ     $0D
LF            equ     $0A

              org     $1000
; the following 7 instructions set up SC0 interrupt jump vector
              ldd     #sc0_isr
              pshd
              ldab    #sc0_vec_no
              clra
              ldx     setuservector
              jsr     0,x
              leas    2,sp

              jsr     init_sc0
              ldx     #msg
              jsr     puts_sc0
              swi
msg           db      "This is testing message!",CR,LF,0
```

```
; ***********************************************************
; The init_sc0 routine configures the SC0 channel with 9600 baud, no parity,
; enable transmit and receive, select long idle mode.
; ***********************************************************
init_sc0    movb    #$00,SC0BDH
            movb    #52,SC0BDL
            movb    #$0C,SC0CR1
            movb    #$0C,SC0CR2
            rts
```

Homework #8

- See course website: http://390.revan.us
  - click homework tab
- Please submit a hard copy