Module 14 – Peripheral communications

Overview

Communications in the computer world takes place in two dimensions. The first is the communication between man and machine for example through a keyboard as previously discussed. The second involves the transfer of data from machine to machine. This is often known as data communications or peripheral communications.

Peripheral communications (concept map)

Objectives

At the completion of this module you will be able to:

- list the concepts of typical communication channels and their major characteristics for digital information transmission
- describe the use of typical interface devices including the Universal Asynchronous Receiver Transmitter (UART)
- demonstrate the basics of programming the Motorola MC68HC11 to transmit and receive data
- state the protocols involved with typical standard interfaces including the RS232 and GPIB systems
- list the techniques involved in long distance digital communication including the operation of a modem.

14.1 Introduction

The link between the machines will depend in many cases on the equipment types and the application. It is often referred to as the communication channel.

14.1.1 Communication channels

Currently communication channels are constructed of copper wire links, coaxial cables, fibre-optic cables or microwave links.

Copper wire links may be composed of twin wire transmission lines (as in a telephone line) or 8, 16, 32, 64 wires grouped together constituting a data bus. These lines can easily be strung from point to point however they are susceptible to external electrical interference from nearby power switching systems, lightening and other sources. All of these can corrupt the data on the lines.

Coaxial cables are twin wire transmission lines constructed so one wire is a tube with the other wire fixed concentrically within the tube. It immediately has the advantages of physical strength together with shielding provided by the exterior tube. Electrical interference to signals on this line is minimised by the shield. It is used as a communication channel within equipment, between equipments and between countries as it can conveniently be laid on the ocean floor or underground. Over long distances the losses in this cable become significant, however they can be compensated for by adding regular repeaters (amplifiers) to the line at appropriate intervals.

Fibre optic cables are a relatively new technology and they provide a communications channel in which the signals are transmitted by light energy instead of electrical energy. This is an extremely efficient, low loss medium which may eventually replace conventional wire based channels. It has other advantages such as lightness of weight, low cost, small size and large carrying capacity. It is frequently being used on new installations throughout the world. A fibre optic cable is a glass rod about the diameter of a human hair. In appropriate lengths it is reasonably flexible and it is supported within a plastic tube for strength in use. Electrical data signals are converted to light pulses at the sending end of the link and back to electrical pulses at the receiving end.

Microwave links are radio transmission systems operating at very high frequencies (4-6GHz) onto which the data signals are modulated. It normally only provides line-of-sight channels and repeaters are used about every 30 miles on the earth to extend the range to any length. Microwave systems can be also used to reach satellites in space to achieve global communications.

A table showing the relative carrying capacities of the transmission channels discussed is now shown.

Channel	Bandwidth	Relative Capacity
Twin Wire	10 – 50000Hz	12
Coaxial Cable	1 – 12MHz	3000
Microwave	1 – 40GHz	50000
Fibre Optics	100 – 1000THz	250000000000

14.1.2 Channel characteristics

Communication channels can be categorised by several characteristics, including the nature of the signal together with the mode, direction and speed of transmission.

Signals on communication channels may be analog or digital. Most conventional long line systems have been designed for analog signals and in many cases the digital signals require converting to analog signals prior to transmission. New digital systems are now becoming available however to improve this situation.

The transmission mode used may be either ‘parallel’ or ‘serial’.

(a) Parallel transmission mode

Short distance communications between machines may take place across several lines simultaneously such as a bus. This is known as parallel transmission. If for instance we wished to transmit the ASCII character 'C' (1000011) we would need 7 lines, one for each binary bit. Other control bus lines would also be required in this system to communicate with the peripheral prior to transmission. This method of transmission is very fast provided the many lines involved are always available. Long distance communication using this mode is prone to interference and expensive as it uses so many lines. It is generally restricted to short distance communications for these reasons.

This course will concentrate on the most widely used method of data communications, which is the serial transmission mode.

(b) Serial transmission mode

Data may be transmitted over a single line in serial form. This means the 7 bits for our ASCII character are transmitted one at a time. However the receiver requires more information than just the data. Otherwise it won't know when one character ends and another starts. In other words the two machines require synchronising. Synchronisation requires two parameters:

- bit synchronisation
- character synchronisation.

Bit synchronisation

Bit synchronisation is necessary so that the receiving device knows when to sample the received signal to see if it is a logic 1 or a logic 0. This is obviously related to the speed at which data is transmitted and correct synchronisation is achieved when both the transmitter and the receiver are operating at identical speeds. This is accomplished by using timing oscillators (clocks) at each end running at identical speeds. The clock at the receiver determines that the data is to be sampled only at intervals corresponding to the frequency of the transmitting clock.

Character synchronisation

Once the receiver and transmitter are operating at identical speeds, the receiver needs to know which bits of the signal make up each character being sent. This is achieved by `framing' the data front and back with more information known as the character synchronisation bits.

There are two methods used frequently to achieve this, serial asynchronous and serial synchronous.

(c) Serial asynchronous

In this transmission mode individual characters (made up of several bits) are transmitted at irregular intervals. That is, as they are entered by a user for example. Characters may be of any pre-defined length however they must be framed by 'start' and 'stop' bits of extra data. The start bits tell the receiver a character is coming and to start sampling and storing the next predefined number of bits as data belonging to a character. The stop bits reconfigure the system in readiness for the next characters arrival. An additional bit known as a parity bit is usually included in the signal for error checking purposes. The period in between characters, which may be of random length is always filled up with logic 1's. It signifies a 'no signal' or 'idle' line condition. A diagram showing an asynchronous signal format is now provided.

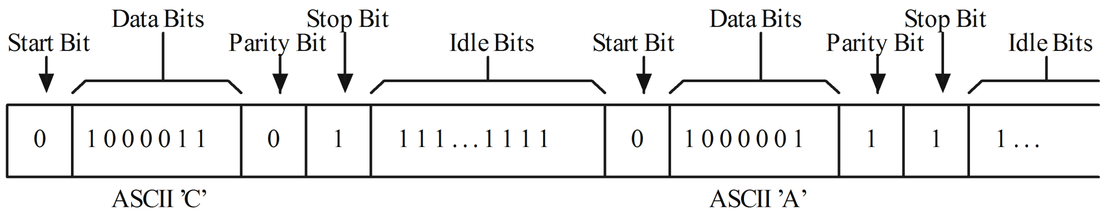


Figure 14.1: Asynchronous transmission format

This mode is used for lower speed data transmission however it is used extensively with most communication equipment designed for personal computers and other microprocessor systems.

(d) Serial synchronous

This mode differs from the asynchronous mode in that large **blocks of data** are transferred at **regular** intervals. In this case the data blocks are framed with synchronising information rather than each character within a block. The receiver becomes synchronised with the transmitter when it detects in the idle stream a special character known as SYN. Once it detects one or two (or more) of these SYN characters it knows that a data block will be coming soon. Preceding the data block is another character called STX (start of text). Once this is received the receiver knows all the next signals constitute data. This is received and stored as pre-defined length words until the ETX (end of text) signal is received. The system then idles until the next block is transmitted. Error checking parity bits are also included in the block. A typical format is now shown.

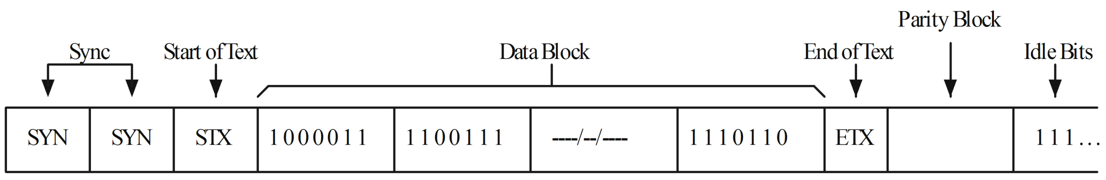


Figure 14.2: Synchronous transmission format

Actually more framing characters involving buffer sizes and addresses can be included in the framing protocol.

This system requires more complex and expensive equipment than the asynchronous mode however it does give higher speeds and greater accuracy.

14.1.3 Direction of transmission

Transmission may be achieved in any one of three general categories. Simplex, half-duplex or full-duplex.

- Simplex transmission means that data flows only in one direction. An example would be a sensor connected to a process. The sensor transmits data relating to that process only and no information is ever sent back to the sensor. A thermistor measuring oil temperature of an engine would be a typical example.

- Half-duplex transmission means data can flow in both directions but only one direction at a time. An example of this is a CB radio system. Users can either talk or listen but not do both simultaneously. Half-duplex is often used between a computer and a peripheral device. Senders must advise when they have finished transmitting and are ready to receive usually by some coded message. In this system the entire bandwidth of the channel is available in either direction so data rates can be high.
- Full-duplex transmission means data can flow in both directions simultaneously. A normal telephone line is an example of this type of channel. This facility can be used for interactive computer applications. A terminal for instance, can interrupt the computer while it is transmitting and ask for other data and also a computer could interrupt a terminal and give it a higher priority task.

Full-duplex systems use half the available channel bandwidth in either direction which usually affects the allowable transmission speed.

14.1.4 Speed of transmission

The rate at which transmission takes place of data on a channel is called **the baud** rate. It is a measure of the number of signal-events per second on the channel. Usually it is used to describe the number of bits sent per second. If a character has 1 start bit + 7 data bits + 1 parity bit + 2 stop bits it means we have 11 bits/character. For a data rate of 10 characters per second we say the transmission speed is 10 x 11 = 110 BAUD. Common baud rates are 110, 150, 300, 600, 1200, 2400, 4800, 9600 and 19200.

Self assessment 14.1

1. Describe using formatted sketches the difference between synchronous and asynchronous serial transmission systems.
2. Sketch and explain a typical signal format used in serial asynchronous data communications.
3. Explain the difference between ‘serial’ and ‘parallel’ transmission modes in connection with a digital communications channel.
4. Explain, using typical examples, the following methods of data transmission.
   i. Simplex
   ii. Half-duplex
   iii. Full-duplex

14.2 Interface devices

Many peripherals connected to computers are in fact serial devices, such as printers, and they require asynchronous serial data transmission.

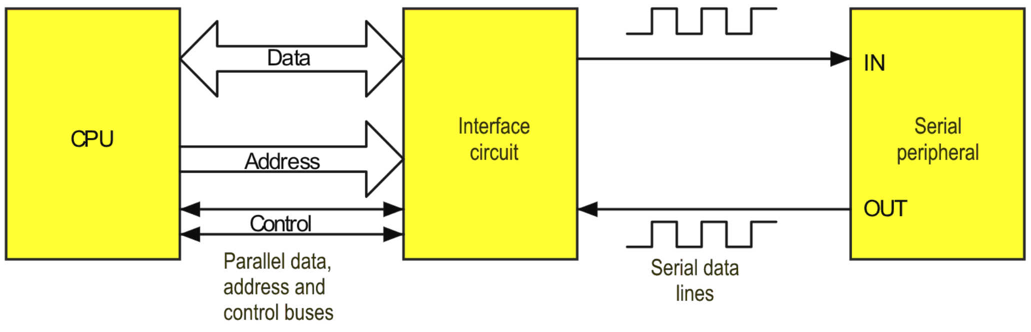


Figure 14.3: Serial interface

This system:

- takes an 8 bit parallel word from the CPU and converts it to a serial data word for the peripheral
- takes a serial data word from the peripheral and converts it to an 8 bit parallel word for the CPU.

It acts as a parallel-to-serial converter and vice versa.

14.2.1 Parallel to serial interfaces

Because of extensive use of this type of communication, manufacturers use a variety of methods to do the 'interface circuit' task shown in the previous diagram.

One method is to use a dedicated LSI device such as a Universal Asynchronous Receiver Transmitter (UART, Motorola 6850) between the CPU and the peripheral. Alternatively, the 'interface circuit' may be integrated onto the same chip as the CPU, e.g. the '68HC11 micro-controller.

The UART

All UARTs are basically identical in nature and they contain:

- A serial receiver (Rx), which takes serial input (Rx DATA) and converts it to a parallel format and stores it in a Rx register (RxDR).
- A serial transmitter (Tx) to convert the parallel word (TxDR) to serial output (TxDATA).
- A bidirectional data bus buffer.
- External clock inputs RxCLK and TxCLK.

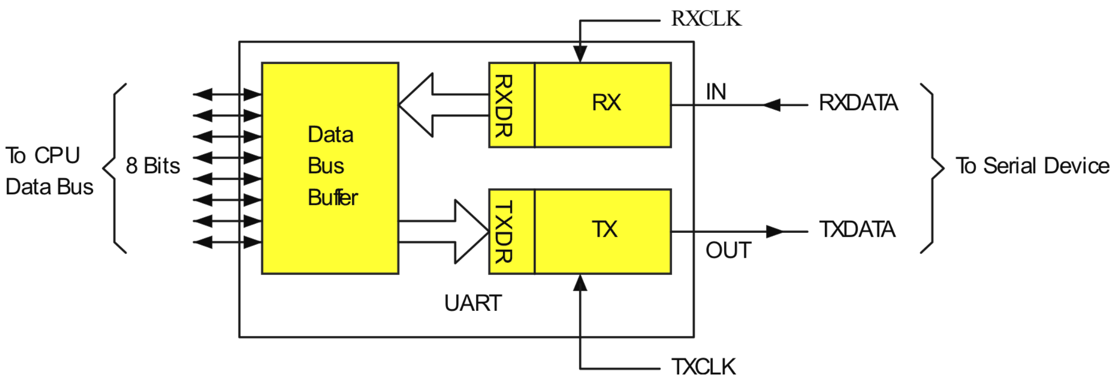


Figure 14.4: Typical UART

In operation, when the CPU wants to transmit a data word to the device it sends the word in on the data bus. This is stored in TxDR.

The Tx section then adds a start bit, parity and stop bits and puts the completed word into a shift register.

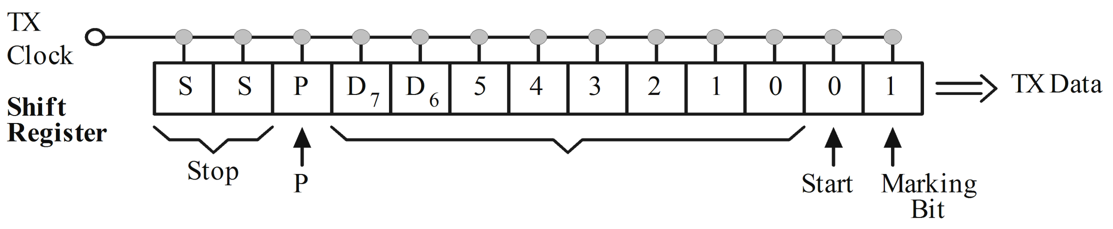


Figure 14.5: Tx shift register

The contents of this register are then shifted right at a rate determined by the TxCLK which has a frequency equal to the desired baud rate. This produces the serial data word output TxDATA.

For reception, the Rx section takes in the device word and puts the data on the data bus for the CPU to read just like it was a memory location. The UART has an internal control register which the CPU can write into also just like a memory location. This register controls the internal functions of the UART.

Synchronising the Rx to the serial data input

As we noted before, the start bit synchronises the Rx to the incoming data. It then knows data, parity and stop bits will follow.

In order to ensure this synchronisation is almost foolproof and that some noise pulse 'glitch' on the line might not be interpreted as a start bit, the UART uses an external clock, which is a much higher frequency than the baud rate, usually 16 times higher. That is, for a rate of 110 baud we have RxCLK = 16 x 110 = 1760 Hz. This is the actual frequency applied to the UART's RxCLK.

In operation, after it senses the first negative transition of the first start bit, the receiver waits 8 periods of RxCLK and then samples the serial input again to see if it is still low. This means it looks about the middle of each bit time T_B. It continues this process of sampling about the centre of each T_B period continuously.

After sampling each bit it stores these samples in a register driven by the Rx data CLK signal derived from the RxCLK.

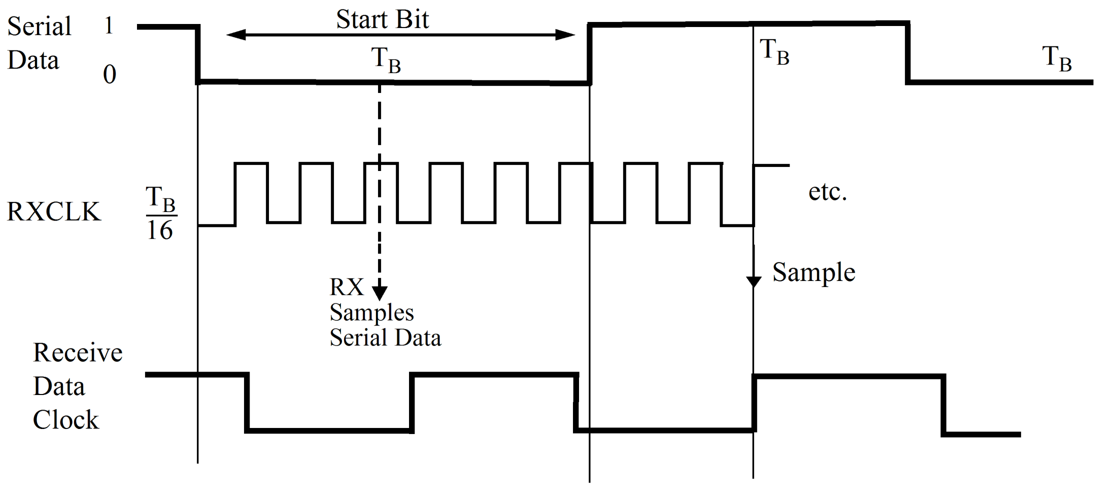


Figure 14.6: Serial timing diagram

Once the complete data word has been shifted into the Rx shift register, circuits in the UART automatically check to see that the correct number of stop bits are logic 1's. If any stop bit = 0, a **framing error** has occurred and a flag is set in an internal UART status register.

This status register can be read by the CPU just like a memory location and CPU can then take appropriate action.

The UART also checks parity and can set a **parity error** flag if an error is detected. Once all the bits are in the RxDR its contents can be transferred to the CPU. This must be done before the next word is put into RxDR or another error called **overrun error** can occur. Again a flag is set in the UART status register.

Overrun errors can be avoided by ensuring the CPU reads the RxDR within the time it takes for a new word to be shifted in. This time of course depends upon the baud rate being used.

14.2.2 The MC68HC11 Micro-controller

The '68HC11 supports both asynchronous and synchronous serial communications by multiplexing the functions on Port D. The 'Serial Communications Interface' (SCI) provides asynchronous communications via pins PD0/RxD and PD1/TxD when enabled. The 'Serial Peripheral Interface' (SPI) provides synchronous communications via pins PD2/MISO, PD3/MOSI, PD4/SCK and PD5/SS when enabled. For the purpose of this course discussion will be limited to a brief overview of the SCI, with registers set for 'normal' operation. You are not expected to recall the purpose of individual bits within each register, but you need to understand how the SCI, in particular, is configured for serial communication.

The SCI consists of 5 registers each with a unique memory address, as follows.

- BAUD – Baud rate control register
- SCCR1 – Control register 1 for data format
- SCCR2 – Control register 2 for mode of operation
- SCSR – Status register generates interrupts and detects data errors
- SCDR – Receive and transmit data register.

Baud

	Bit 7	6	5	4	3	2	1	0
$002B	TCLR	0	SCP1	SCP0	RCKB	SCR2	SCR1	SCR0

This register controls the baud rate. SCP1 and SCP0 control the baud rate pre-scalar, which consists of 4 clock frequency 'division factors' (DF) – 1, 3, 4 and 13 that correspond to the four possible settings of SCP1 and SCP0, i.e. 00, 01, 10, 11 respectively.

Example

If we assume a clock frequency of 1MHz and an over sampling rate of 16 then the highest baud rate possible is 62.5Kbaud. Over sampling refers to the number of times the received bit is sampled by the hardware in an effort to reduce errors.

Setting SCP1 and SCP0 to 00, DF = 1 and the pre-scalar baud rate is 62.5Kbaud.

Setting SCP1 and SCP0 to 11, DF = 13 and the pre-scalar baud rate is 4800baud.

The pre-scalar baud rates are further divided based on the values of SCR2, SCR1 and SCR0. That is, a further 8 division factors – 1,2,4,8,16,32,64 and 128, corresponding to 000, 001, 010, etc.

Example

Using the previous example, the pre-scalar DF is set to 13 (4800baud). If SCR2, SCR1 and SCR0 are set to 010 (DF = 4) then the final baud rate is 1200baud.

Ignoring the TCLR and RCKB bits, the required BAUD register ($002B) word for 1200baud is $32.

SCCR1

$002C	R8	T8	0	M	WAKE	0	0	0

This register controls the data format. The character length is set by the M bit as follows:

M = 0 – one start bit, 8 data bits, one stop bit
M = 1 – one start bit, 9 data bits, one stop bit
The additional data bit can be used for parity checking or an extra stop bit.
M = 0 is the most common setting. With this setting the other bits can be ignored.

So, for normal operation the required SCCR1 register ($002C) word for one start bit, 8 data bits, one stop bit is $00.

$002D	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK

SCCR2

$002D	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK

This register controls the mode of operation. TIE is the transmit interrupt enable bit. This bit is set to 0 for software polling of the status register bit TDRE.

TCIE is the transmit complete interrupt enable. Again, this bit is set to 0 for software polling of the status register TC bit.

RIE is the receiver interrupt enable. When a character is received and transferred into the RDR, we need to get an RDRF interrupt from the status register. Therefore this bit is set to 1.

ILIE is the idle interrupt enable and can be set to 0 to disable idle interrupts.

TE is the transmitter enable bit and is set to 1 to enable the SCI transmitter.

RE is the receiver enable bit and is set to 1 to enable the SCI receiver.

RWU (receiver wake up) is set to 0 for normal receiver operation.

SBK (transmitter send break character) is set to 0 for normal operation.

So, for normal operation the required SCCR2 register ($002D) word is $2C.

SCSR

$002E	TDRE	TC	RDRF	IDLE	OR	NF	FE	0

This register generates interrupts and detects data errors. TDRE is the transmit data register empty bit. This bit will be set (1) by the SCI hardware to indicate that a new character can be written to the SCDR. An interrupt will not be generated as software polling of this bit is used for normal operation.

RDRF is the receive data register full bit, and is the only other bit of interest in this register. This bit will be set and an interrupt generated to indicate that a character has been received and has transferred from the receive shift register to the SCDR where software can read it. An interrupt service routine will be required to read the data.

SCDR

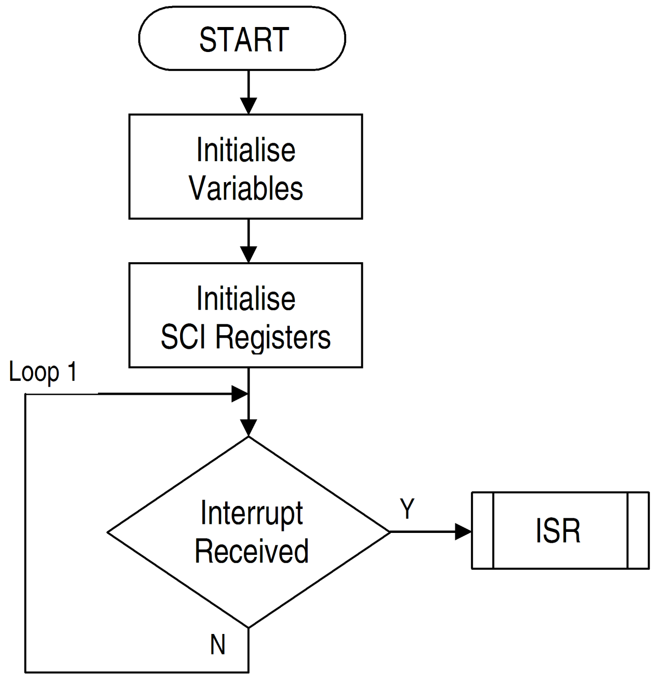
$002F	R7	R6	R5	R4	R3	R2	R1	R0
	T7	T6	T5	T4	T3	T2	T1	T0

This register is used to receive and transmit data.

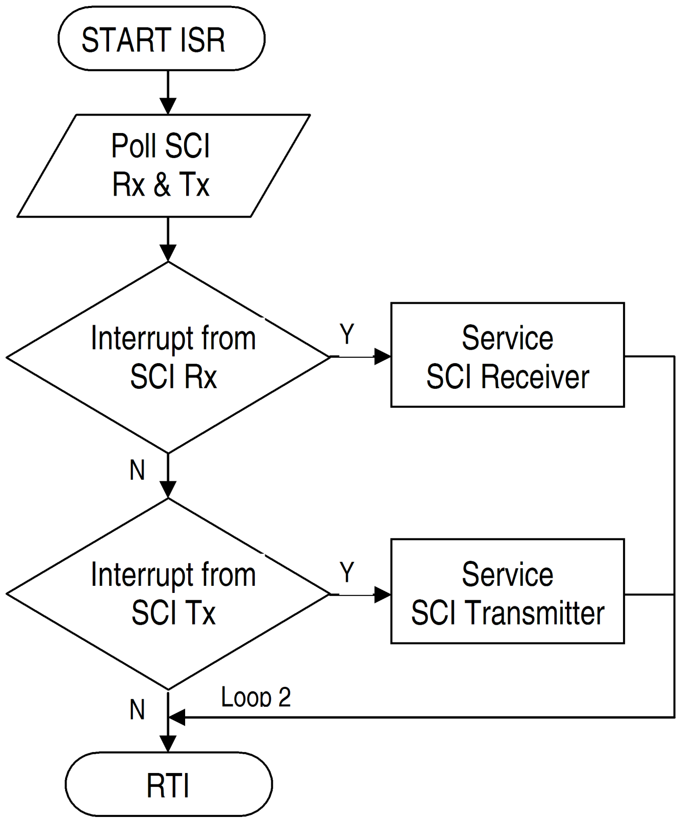
To use the Serial Communications Interface (SCI) refer to home experiments 14.1 and 14.2.

14.2.3 Program example (flow chart only)

Main program:



Interrupt service routine:



Activity 14.1

a. Refer to the ELE1301 course page on 'Study Desk' and complete the following experiments:

  iv. Home experiment 14.1 – SCI Polling

  v. Home expierment 14.2 – SCI Interrupts

Self assessment 14.2

1. Define baud rate and indicate how it could be determined for the following data stream; 1 start bit + 8 data bits + 1 stop bit, @ 30 characters per second.
2. Draw a properly formatted flow chart for an interrupt driven '68HC11 Serial Communications Interface.
3. Sketch a block diagram of a typical asynchronous parallel to serial interface and explain the method used to transmit and receive serial data.
4. Explain, with the aid of a timing diagram, the method used to synchronise the receiver to the serial data input.
5. Define the following error terms associated with serial data transfer:
   i. Framing error
   ii. Parity error
   iii. Overrun error

14.3 Standard interfaces

There are 2 basic interfaces in use today:

- Electronic Industries Association RS-232 interface
- IEEE STD 488 interface (also known as the Hewlett Packard interface on the General Purpose Interface Bus, GPIB).

14.3.1 The RS-232 interface

This standard specifies signal voltage levels and handshake signals between data terminal equipment and data communications equipment.

Typically RS232-C uses voltages -12 for a logic 1 and +12 for a logic 0.

Most standard IC's are TTL and use 0, and +5 V for logic 0 and 1 respectively. This requires some interface circuitry to make these voltage levels to be compatible.

The RS-232 system has many other features. For instance it is a **standard** which sets out which signals should appear on which pins of a standard 25 pin multi-way plug and cable as shown in the diagram on the following page.

The signals are divided into: data, control and timing.

Few installations use all of the signals available on the systems 25 wires. The standard defines that each wire can convey one of two voltages according to the following table:

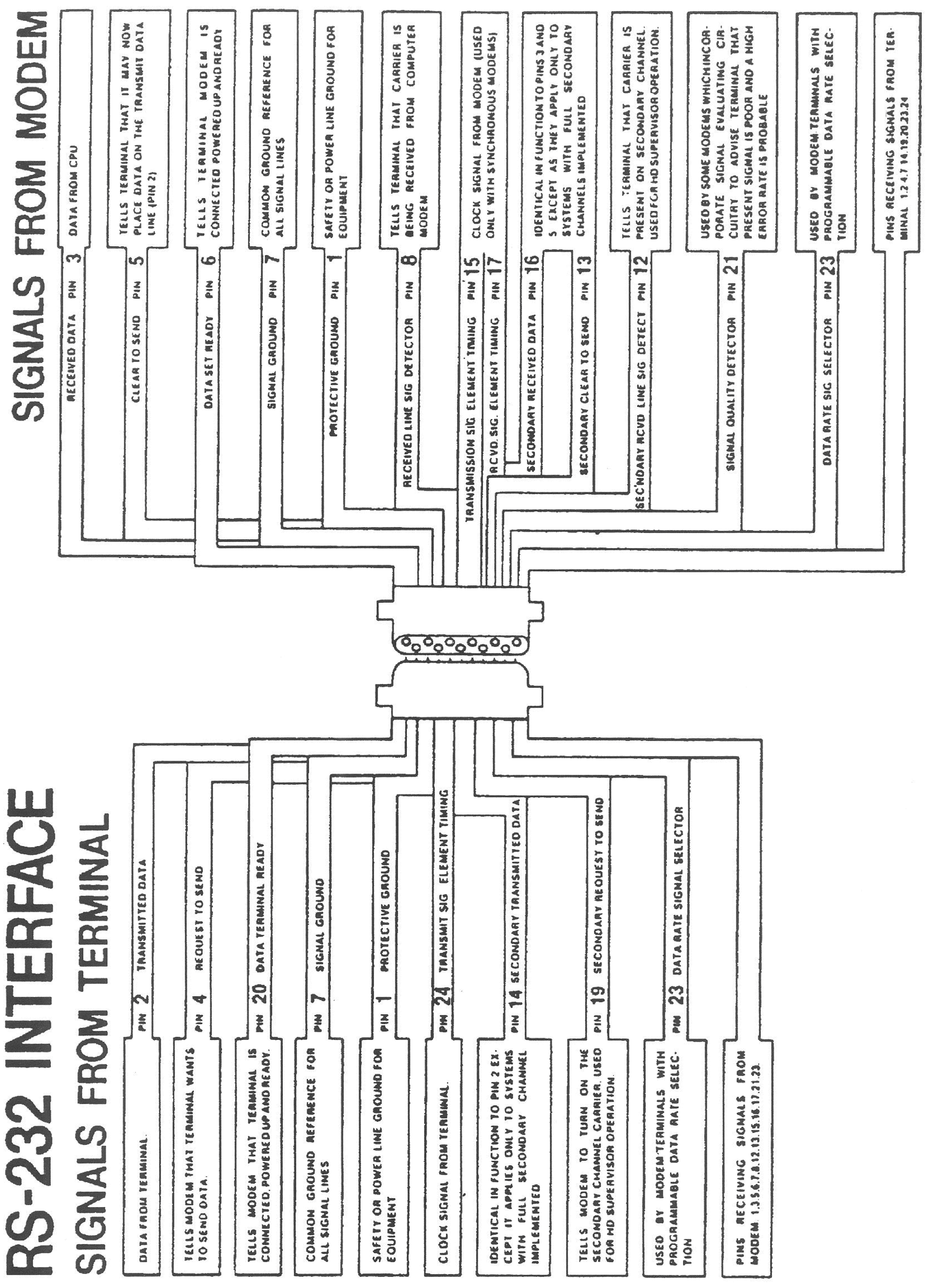
Bin	Voltage
ON, 0, SPACE	+3 to +25 V
OFF, 1, MARK	-3 to -25 V

Most systems use a standard set of lines as shown in the next table:

Table 14.1: Commonly-used RS-232 signals

Signal name	Mnemonic	Pin	Use
Protective Ground	GND	1	A connection to the terminals metal chassis.
Transmitted Data	TDATA	2	Outgoing data path from the terminals point of view.
Received Data	RDATA	3	Incoming data path from the terminals point of view.
Request to Send	RTS	4	Activated by the terminal to tell the modem to prepare to receive data from the terminal.
Clear to Send	CTS	5	Activated by the modem to tell the terminal that it is ready to receive data from the terminal.
Data Set Ready	DSR	6	Activated by the modem to tell the terminal that the modem is operational.
Signal Ground	SGND	7	Return path for all other signals on the bus.
Received Line Signal Detector	RLSD	8	Activated by the modem to tell the terminal that the modem has made contact with the computer and can sense the carrier.
Data Terminal Ready	DTR	20	Activated by the terminal to tell the modem that the terminal is operational.

Note: 2 devices each with the same RS-232 standard may still be unable to communicate. Other parameters such as **timing** and **format variables** are critical for communications and are not covered by the standard. Fortunately most manufacturers are aware of this and try to make their systems compatible with others.


RS-232 INTERFACE
SIGNALS FROM TERMINAL
DATA FROM TERMINAL. PIN 2 TRANSMITTED DATA
TELLS MODEM THAT TERMINAL WANTS TO SEND DATA. PIN 4 REQUEST TO SEND
TELLS MODEM THAT TERMINAL IS CONNECTED, POWERED UP AND READY. PIN 20 DATA TERMINAL READY
COMMON GROUND REFERENCE FOR ALL SIGNAL LINES PIN 7 SIGNAL GROUND
SAFETY OR POWER LINE GROUND FOR EQUIPMENT PIN 1 PROTECTIVE GROUND
CLOCK SIGNAL FROM TERMINAL. PIN 24 TRANSMIT SIG ELEMENT TIMING
IDENTICAL IN FUNCTION TO PIN 2 EXCEPT IT APPLIES ONLY TO SYSTEMS WITH FULL SECONDARY CHANNEL IMPLEMENTED PIN 14 SECONDARY TRANSMITTED DATA
TELLS MODEM TO TURN ON THE SECONDARY CHANNEL CARRIER. USED FOR HD SUPERVISOR OPERATION. PIN 19 SECONDARY REQUEST TO SEND
USED BY MODEM TERMINALS WITH PROGRAMMABLE DATA RATE SELECTION PIN 23 DATA RATE SIGNAL SELECTOR
PINS RECEIVING SIGNALS FROM MODEM 1,3,5,6,7,8,12,13,15,16,17,21,23.
SIGNALS FROM MODEM
RECEIVED DATA PIN 3 DATA FROM CPU
CLEAR TO SEND PIN 5 TELLS TERMINAL THAT IT MAY NOW PLACE DATA ON THE TRANSMIT DATA LINE (PIN 2)
DATA SET READY PIN 6 TELLS TERMINAL MODEM IS CONNECTED POWERED UP AND READY
SIGNAL GROUND PIN 7 COMMON GROUND REFERENCE FOR ALL SIGNAL LINES
PROTECTIVE GROUND PIN 1 SAFETY OR POWER LINE GROUND FOR EQUIPMENT
RECEIVED LINE SIG DETECTOR PIN 8 TELLS TERMINAL THAT CARRIER IS BEING RECEIVED FROM COMPUTER MODEM
TRANSMISSION SIG ELEMENT TIMING PIN 15
RCVD. SIG. ELEMENT TIMING PIN 17
CLOCK SIGNAL FROM MODEM (USED ONLY WITH SYNCHRONOUS MODEMS)
SECONDARY RECEIVED DATA PIN 16
SECONDARY CLEAR TO SEND PIN 13
IDENTICAL IN FUNCTION TO PINS 3 AND 5 EXCEPT AS THEY APPLY ONLY TO SYSTEMS WITH FULL SECONDARY CHANNELS IMPLEMENTED
SEC'NDARY RCVD LINE SIG DETECT PIN 12 TELLS TERMINAL THAT CARRIER IS PRESENT ON SECONDARY CHANNEL. USED FOR HD SUPERVISOR OPERATION.
SIGNAL QUALITY DETECTOR PIN 21 USED BY SOME MODEMS WHICH INCORPORATE SIGNAL EVALUATING CIRCUITRY TO ADVISE TERMINAL THAT PRESENT SIGNAL IS POOR AND A HIGH ERROR RATE IS PROBABLE
DATA RATE SIG SELECTOR PIN 23 USED BY MODEM TERMINALS WITH PROGRAMMABLE DATA RATE SELECTION
PINS RECEIVING SIGNALS FROM TERMINAL 1,2,4,7,14,19,20,23,24

When talking about interfaces – as you'll notice in the diagram of RS-232-C in Appendix A – it is important to make the distinction between data sent from the terminal (DTE) and data from the data communications equipment (DCE). DTE refers to the terminal: The 'terminal' may be either a data entry device, a printer or a computer. DCE refer to the data communications equipment (modem).

(Source: Tektronix 1979, *Essentials of data communication*, Tektronix Worldwide Marketing Centres, Beaverton, Oregon, USA, p. 14.)

14.3.2 The IEEE STD 488 or GPIB interface

This system is designed for use in a network of devices usually up to 15. It uses a 24 line bus comprising:

- 8 ground lines
- 8 data lines
- 8 control lines.

A typical bus system is shown in the diagram.

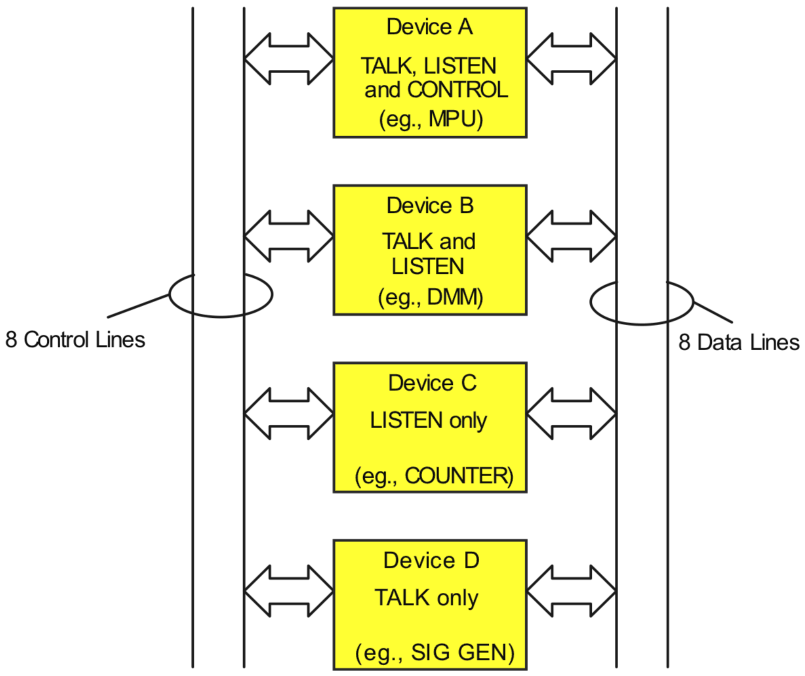


Figure 14.7: GPIB interface

Devices are classified into 3 types:

- talkers that can Tx data on the data lines
- listeners that can Rx data from the data lines
- controllers that designate which devices can talk or listen.

Bus operation

In the control mode, a signal (ATN) is set low so all devices monitor the bus. Commands such as **untalk and unlisten** are then sent to eliminate all previous connection commands LISTEN (ADDRESS) and TALK (ADDRESS) are sent to reconfigure various devices to either talk or listen.

Addresses are assigned by manufacturer and only one device can talk at a time, however several can listen. A device can be addressed as a listener or a talker but not both at once.

If ATN is set high now, then only the previously addressed devices will participate.

IEE-488 signals

The signals on the various lines are described below:

8 Data Lines

These lines are indicated as D10-1 to D10-8 and data flows over them in bit parallel, byte serial form.

3 Data Transfer Lines

These lines provide communication between the talker and the listener. They are:

NDAC	meaning the device is ready to accept information
NRFD	indicating information is accepted by the listener
DAV	indicating that valid data is available on the data lines

5 Management Lines

These lines control the information flow on the bus. They are:

ATN	indicates type of data on the data lines
IFC	configures the system in a desired state
SRQ	request for service
REN	remote operation selection
EOI	indicates completion of a transfer.

Self assessment 14.3

1. Describe one standard interface system with which you are familiar.

14.4 Communications over long distances

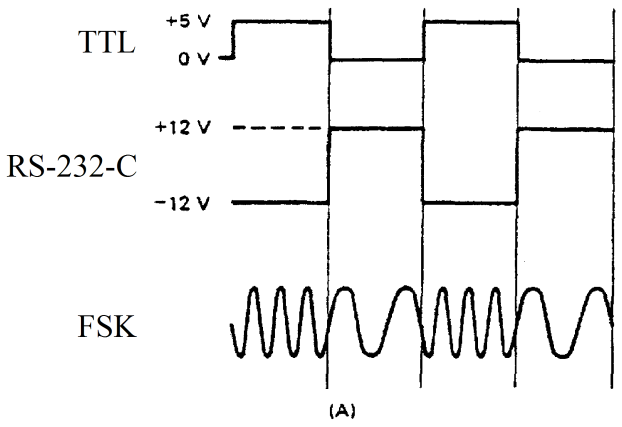
14.4.1 Modems

Computer systems often use the standard telephone lines to send digital data between a terminal and a computer. The available bandwidth is only 3.4 kHz on a standard telephone line.

The bandwidth of a square wave (digital signal) is practically very large and therefore such pulses cannot be transmitted over a conventional telephone line.

One common technique is to convert the digital pulses to sine waves. One frequency for a 0 and another for a 1. This is called Frequency Shift Keying (FSK).

Special circuits are available to convert the logic pulses to sine waves (modulation) and to convert the sine waves back to logic signals again (demodulation). A circuit, which performs both functions is called a modulator/demodulator or **modem**. A typical modem is shown in the diagram.



The modem has 2 sides, RS232 side and the FSK side.

A typical situation where a MPU is communicating with a remote main computer is shown in the diagram.

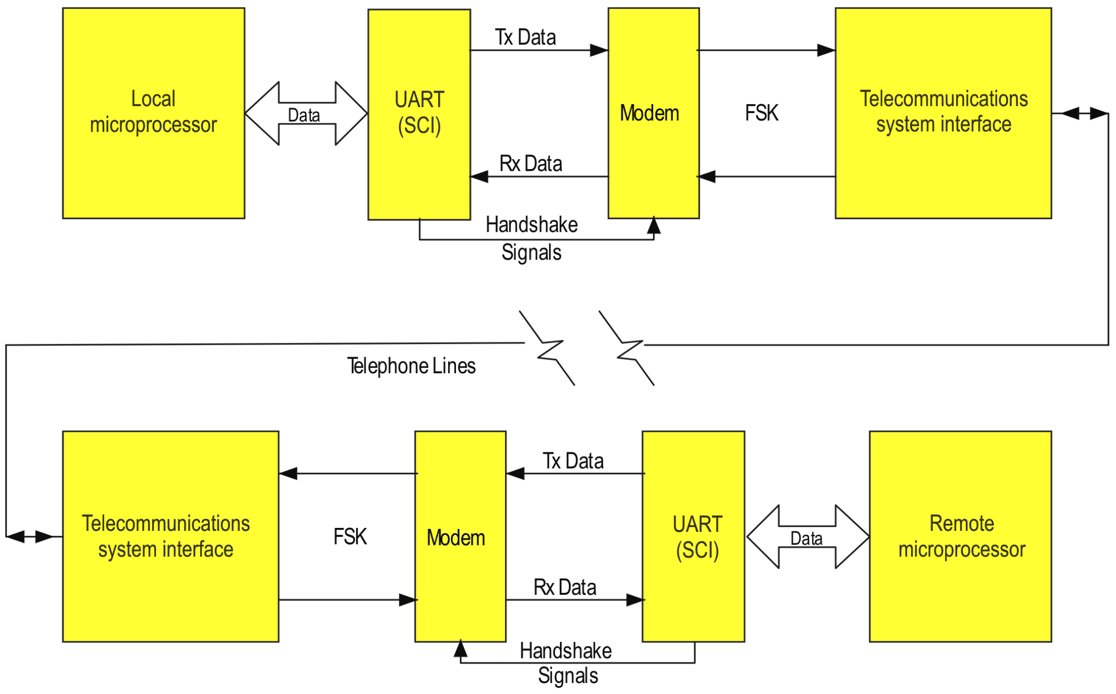


Figure 14.8: Serial communication system

The diagram shows several RS232 lines between the UART and the **modem** for handshaking (communication). RS232 C specifies 6 handshake signals but practically only 2 or 3 are actually used. The most common are:

- Request to Send (RTS) (generated by UART and sent to modem)
- Clear to Send (CTS) (generated by-modem and sent to UART)

14.4.2 Modem frequencies

Typically modems use frequencies:

Mark (Logic 1) = 1270 Hz

Space (Logic 0) = 1070 Hz

for originating modems

and Logic 1 = 2225 Hz

Logic 0 = 2025 Hz for answer modems.

These two allocations of frequencies allow full duplex if required.

Self assessment

1. List four (4) commonly used RS-232 handshaking signals. Give the mnemonic for each and explain their use.