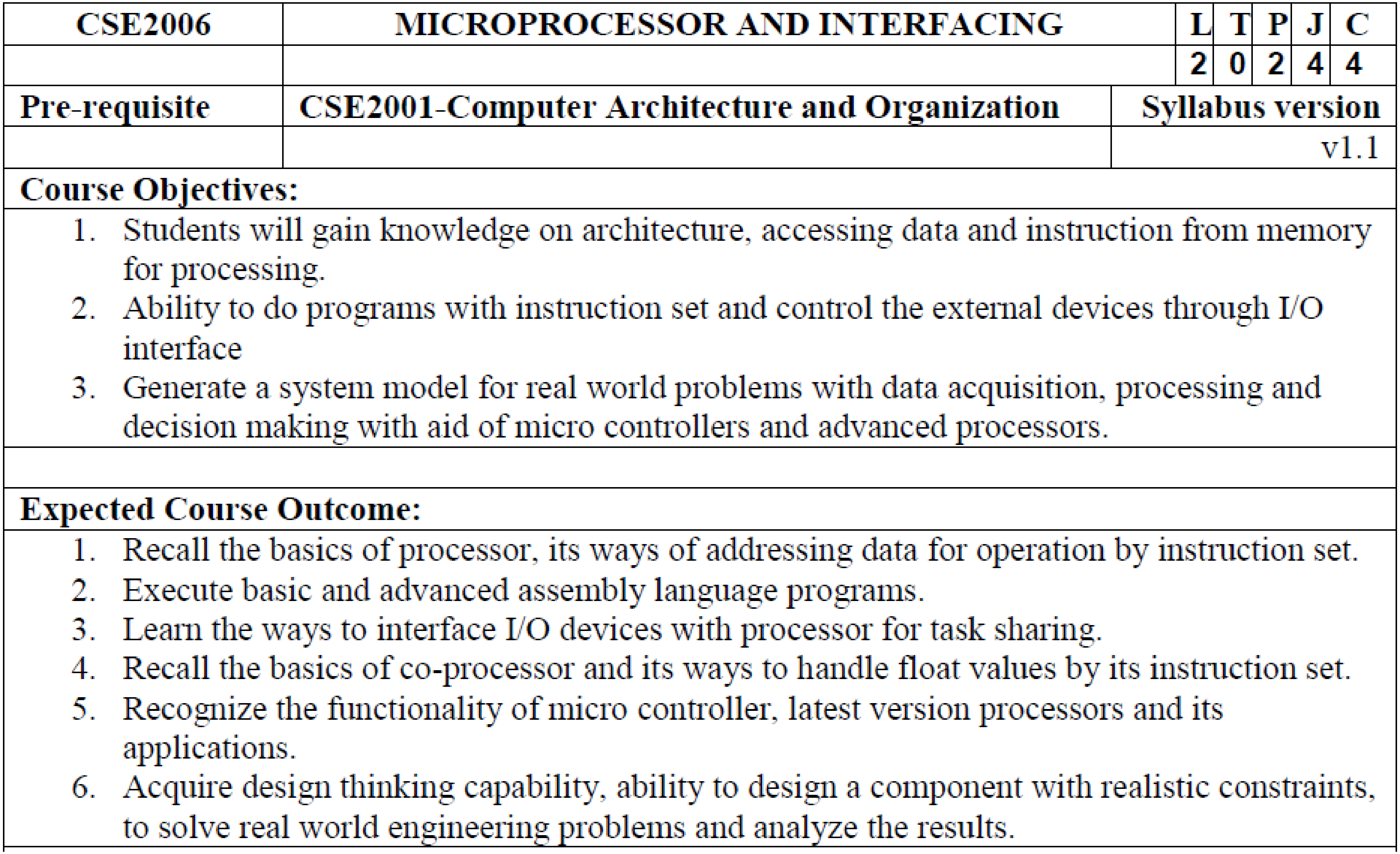
CSE2006
Microprocessor & Interfacing

Module – 5
Introduction to Peripheral Interfacing II

Dr. E. Konguvel
Assistant Professor (Sr. Gr. 1),
Dept. of Embedded Technology,
School of Electronics Engineering (SENSE),
konguvel.e@vit.ac.in
9597812810

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

Syllabus



CSE2006	**MICROPROCESSOR AND INTERFACING**	**L**	**T**	**P**	**J**	**C**
		2	0	2	4	4
Pre-requisite	**CSE2001-Computer Architecture and Organization**	**Syllabus version**				
		v1.1				

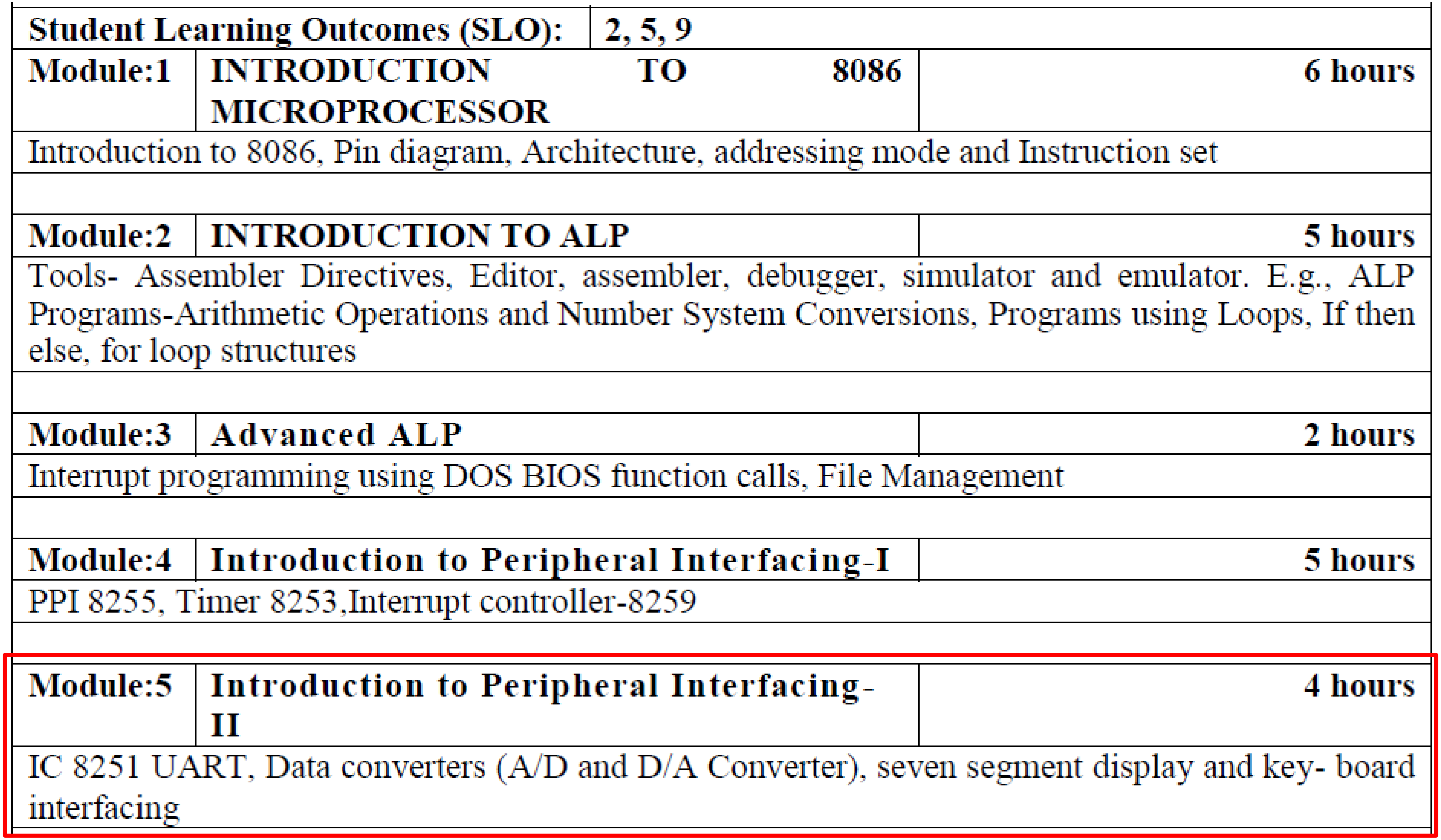
Course Objectives:

1. Students will gain knowledge on architecture, accessing data and instruction from memory for processing.
2. Ability to do programs with instruction set and control the external devices through I/O interface
3. Generate a system model for real world problems with data acquisition, processing and decision making with aid of micro controllers and advanced processors.

Expected Course Outcome:

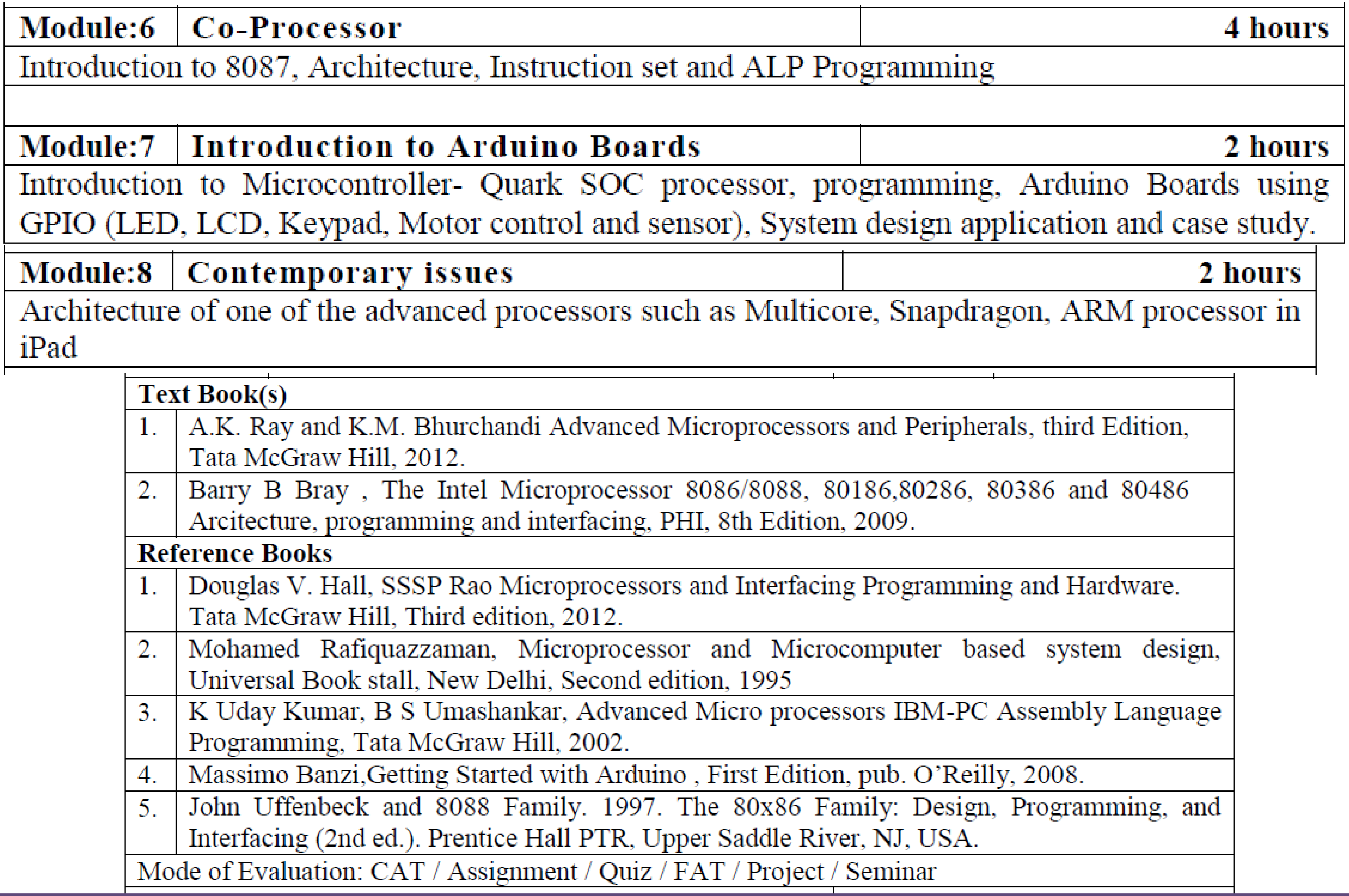
1. Recall the basics of processor, its ways of addressing data for operation by instruction set.
2. Execute basic and advanced assembly language programs.
3. Learn the ways to interface I/O devices with processor for task sharing.
4. Recall the basics of co-processor and its ways to handle float values by its instruction set.
5. Recognize the functionality of micro controller, latest version processors and its applications.
6. Acquire design thinking capability, ability to design a component with realistic constraints, to solve real world engineering problems and analyze the results.

Syllabus



Student Learning Outcomes (SLO):	**2, 5, 9**	
Module:1	**INTRODUCTION TO 8086 MICROPROCESSOR**	**6 hours**
Introduction to 8086, Pin diagram, Architecture, addressing mode and Instruction set		
Module:2	**INTRODUCTION TO ALP**	**5 hours**
Tools- Assembler Directives, Editor, assembler, debugger, simulator and emulator. E.g., ALP Programs-Arithmetic Operations and Number System Conversions, Programs using Loops, If then else, for loop structures		
Module:3	**Advanced ALP**	**2 hours**
Interrupt programming using DOS BIOS function calls, File Management		
Module:4	**Introduction to Peripheral Interfacing-I**	**5 hours**
PPI 8255, Timer 8253,Interrupt controller-8259		
Module:5	**Introduction to Peripheral Interfacing-II**	**4 hours**
IC 8251 UART, Data converters (A/D and D/A Converter), seven segment display and key- board interfacing		

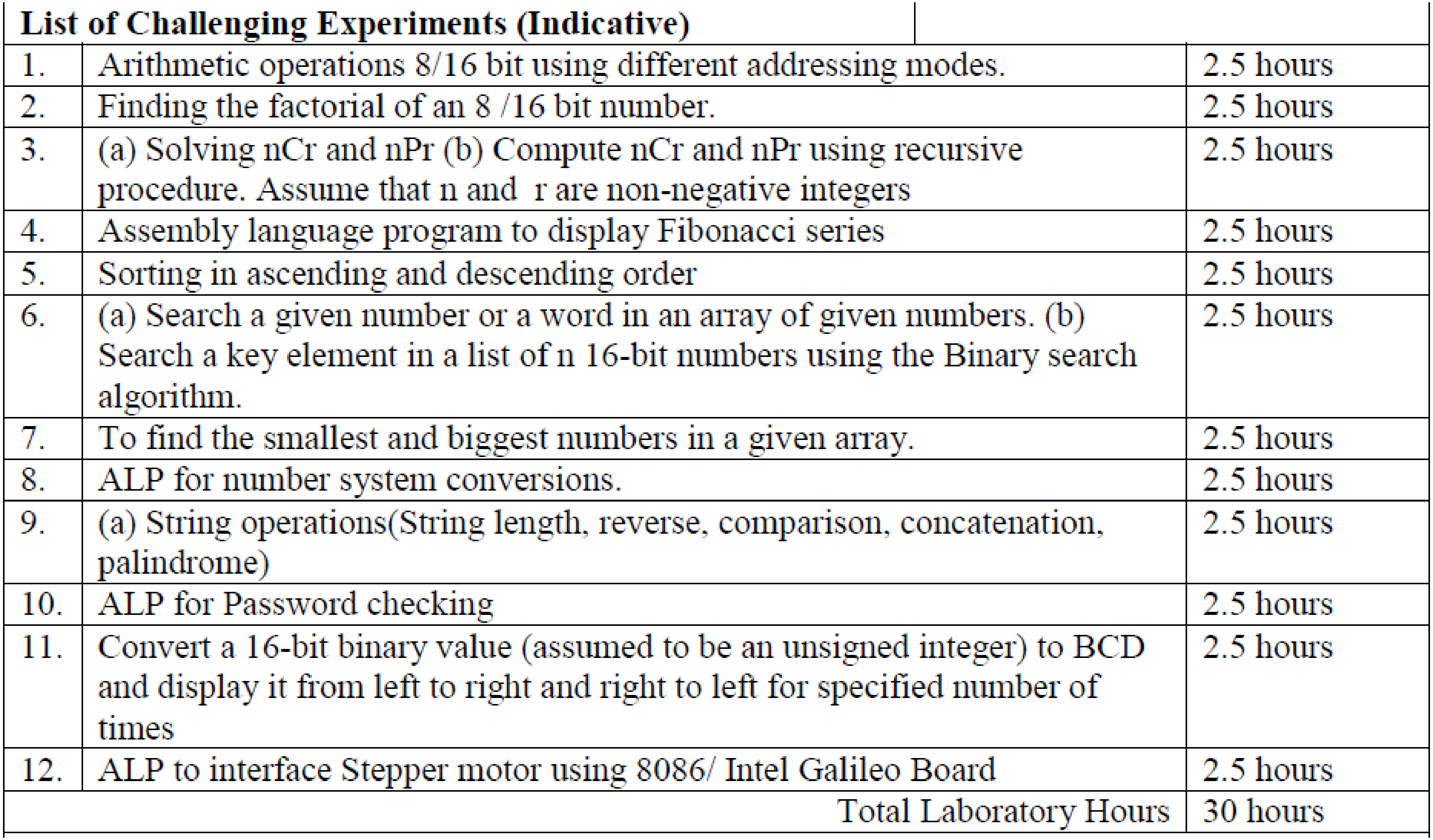
Syllabus



Module:6	**Co-Processor**	**4 hours**
Introduction to 8087, Architecture, Instruction set and ALP Programming		
Module:7	**Introduction to Arduino Boards**	**2 hours**
Introduction to Microcontroller- Quark SOC processor, programming, Arduino Boards using GPIO (LED, LCD, Keypad, Motor control and sensor), System design application and case study.		
Module:8	**Contemporary issues**	**2 hours**
Architecture of one of the advanced processors such as Multicore, Snapdragon, ARM processor in iPad		

Text Book(s)	
1.	A.K. Ray and K.M. Bhurchandi Advanced Microprocessors and Peripherals, third Edition, Tata McGraw Hill, 2012.
2.	Barry B Bray , The Intel Microprocessor 8086/8088, 80186,80286, 80386 and 80486 Arcitecture, programming and interfacing, PHI, 8th Edition, 2009.
Reference Books	
1.	Douglas V. Hall, SSSP Rao Microprocessors and Interfacing Programming and Hardware. Tata McGraw Hill, Third edition, 2012.
2.	Mohamed Rafiquazzaman, Microprocessor and Microcomputer based system design, Universal Book stall, New Delhi, Second edition, 1995
3.	K Uday Kumar, B S Umashankar, Advanced Micro processors IBM-PC Assembly Language Programming, Tata McGraw Hill, 2002.
4.	Massimo Banzi,Getting Started with Arduino , First Edition, pub. O'Reilly, 2008.
5.	John Uffenbeck and 8088 Family. 1997. The 80x86 Family: Design, Programming, and Interfacing (2nd ed.). Prentice Hall PTR, Upper Saddle River, NJ, USA.
Mode of Evaluation: CAT / Assignment / Quiz / FAT / Project / Seminar	

Syllabus



List of Challenging Experiments (Indicative)		
1.	Arithmetic operations 8/16 bit using different addressing modes.	2.5 hours
2.	Finding the factorial of an 8 /16 bit number.	2.5 hours
3.	(a) Solving nCr and nPr (b) Compute nCr and nPr using recursive procedure. Assume that n and r are non-negative integers	2.5 hours
4.	Assembly language program to display Fibonacci series	2.5 hours
5.	Sorting in ascending and descending order	2.5 hours
6.	(a) Search a given number or a word in an array of given numbers. (b) Search a key element in a list of n 16-bit numbers using the Binary search algorithm.	2.5 hours
7.	To find the smallest and biggest numbers in a given array.	2.5 hours
8.	ALP for number system conversions.	2.5 hours
9.	(a) String operations(String length, reverse, comparison, concatenation, palindrome)	2.5 hours
10.	ALP for Password checking	2.5 hours
11.	Convert a 16-bit binary value (assumed to be an unsigned integer) to BCD and display it from left to right and right to left for specified number of times	2.5 hours
12.	ALP to interface Stepper motor using 8086/ Intel Galileo Board	2.5 hours
	Total Laboratory Hours	30 hours

Module 5: Introduction Peripheral Interfacing II

- **Serial Communication Interface – 8251**
- Analog-to-Digital Converter Interfacing
- Digital-to-Analog Converter Interfacing
- Seven segment display interfacing
- Keyboard interfacing

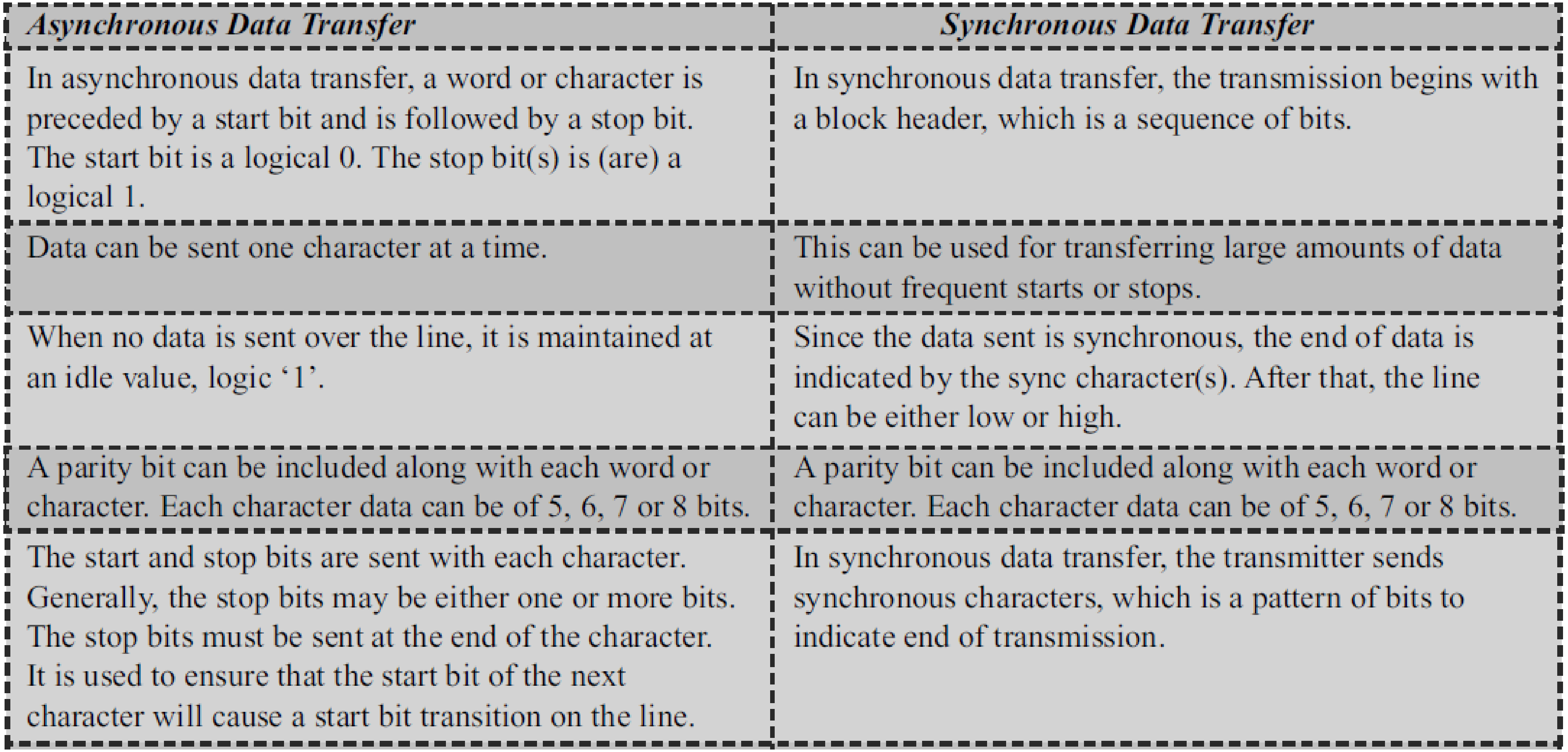
Serial Communication Interface – 8251

- Introduction
- Features
- Functional Diagram
- Pin Diagram
- Interfacing with Microprocessor
- Programming and Operating Modes
- Problems

8251 – Introduction

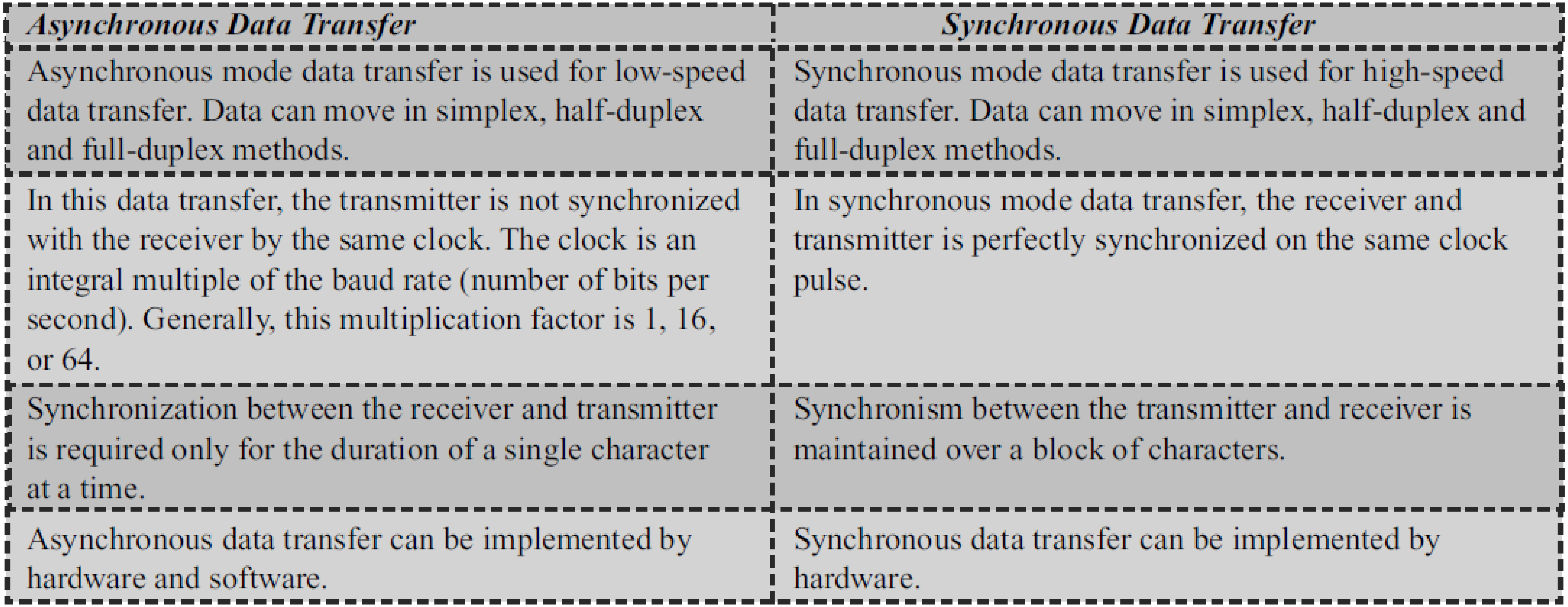
- The serial data transfer is a method of data transfer in which one bit is transferred at a time.
- Used when the distance is greater than five metres, since it requires very few data lines compared to parallel transmission.
- Classified as **simplex** (one direction), **half-duplex** (either direction but in one direction at a time) and **full-duplex** (both directions).
- Classified based on timing signals such as synchronous and asynchronous data transfer.

8251 – Introduction



Asynchronous Data Transfer	*Synchronous Data Transfer*
In asynchronous data transfer, a word or character is preceded by a start bit and is followed by a stop bit. The start bit is a logical 0. The stop bit(s) is (are) a logical 1.	In synchronous data transfer, the transmission begins with a block header, which is a sequence of bits.
Data can be sent one character at a time.	This can be used for transferring large amounts of data without frequent starts or stops.
When no data is sent over the line, it is maintained at an idle value, logic '1'.	Since the data sent is synchronous, the end of data is indicated by the sync character(s). After that, the line can be either low or high.
A parity bit can be included along with each word or character. Each character data can be of 5, 6, 7 or 8 bits.	A parity bit can be included along with each word or character. Each character data can be of 5, 6, 7 or 8 bits.
The start and stop bits are sent with each character. Generally, the stop bits may be either one or more bits. The stop bits must be sent at the end of the character. It is used to ensure that the start bit of the next character will cause a start bit transition on the line.	In synchronous data transfer, the transmitter sends synchronous characters, which is a pattern of bits to indicate end of transmission.

8251 – Introduction



Asynchronous Data Transfer	*Synchronous Data Transfer*
Asynchronous mode data transfer is used for low-speed data transfer. Data can move in simplex, half-duplex and full-duplex methods.	Synchronous mode data transfer is used for high-speed data transfer. Data can move in simplex, half-duplex and full-duplex methods.
In this data transfer, the transmitter is not synchronized with the receiver by the same clock. The clock is an integral multiple of the baud rate (number of bits per second). Generally, this multiplication factor is 1, 16, or 64.	In synchronous mode data transfer, the receiver and transmitter is perfectly synchronized on the same clock pulse.
Synchronization between the receiver and transmitter is required only for the duration of a single character at a time.	Synchronism between the transmitter and receiver is maintained over a block of characters.
Asynchronous data transfer can be implemented by hardware and software.	Synchronous data transfer can be implemented by hardware.

8251 – Introduction

- The 8251 is a powerful programmable communication interface IC through which the serial data transfer can be effectively carried out.

- Used either in synchronous mode or asynchronous mode, so it is called Universal Synchronous Asynchronous Receiver and Transmitter (USART).

- Fabricated using N-channel silicon gate technology.

- Accepts data in parallel/serial data from the microprocessor and converts them into serial/parallel data for transmission.

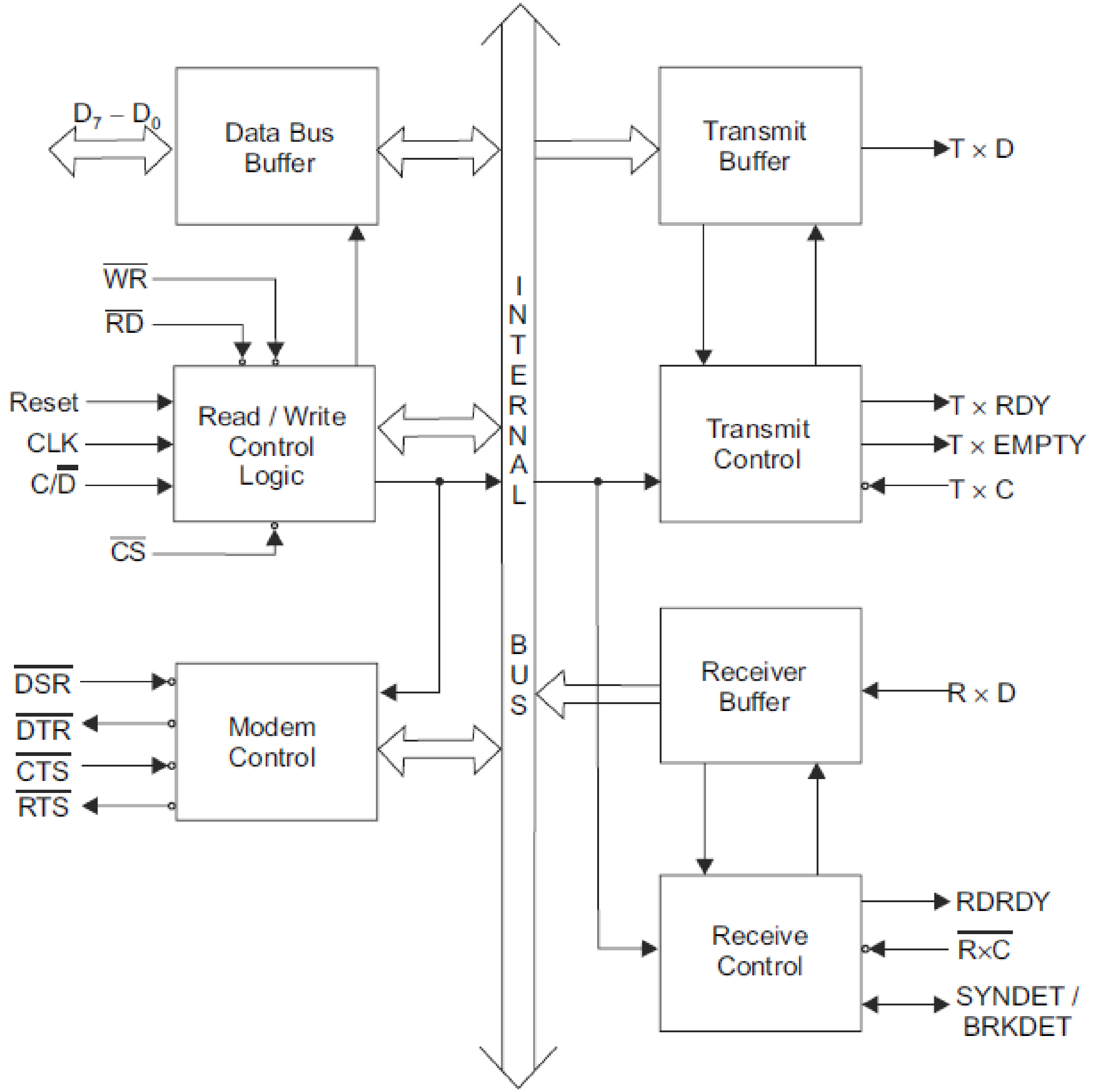
8251 – Features

- Synchronous and asynchronous operation
- Programmable data word length, parity and stop bits
- Parity, overrun and framing error-checking instructions and counting-loop interactions
- Programmed for three different baud rates
- Supports up to 1.750 Mbps transmission rates
- Divide-by 1, 16, 64 mode
- False start bit deletion
- Number of stops increase of asynchronous data transfer can be 1 bit 1 ½ or 2 bits
- Full-duplex double-buffered transmitter and receiver
- Automatic break detection
- Internal and external sync character detection
- Peripheral modem control functions

8251 – Functional Block Diagram

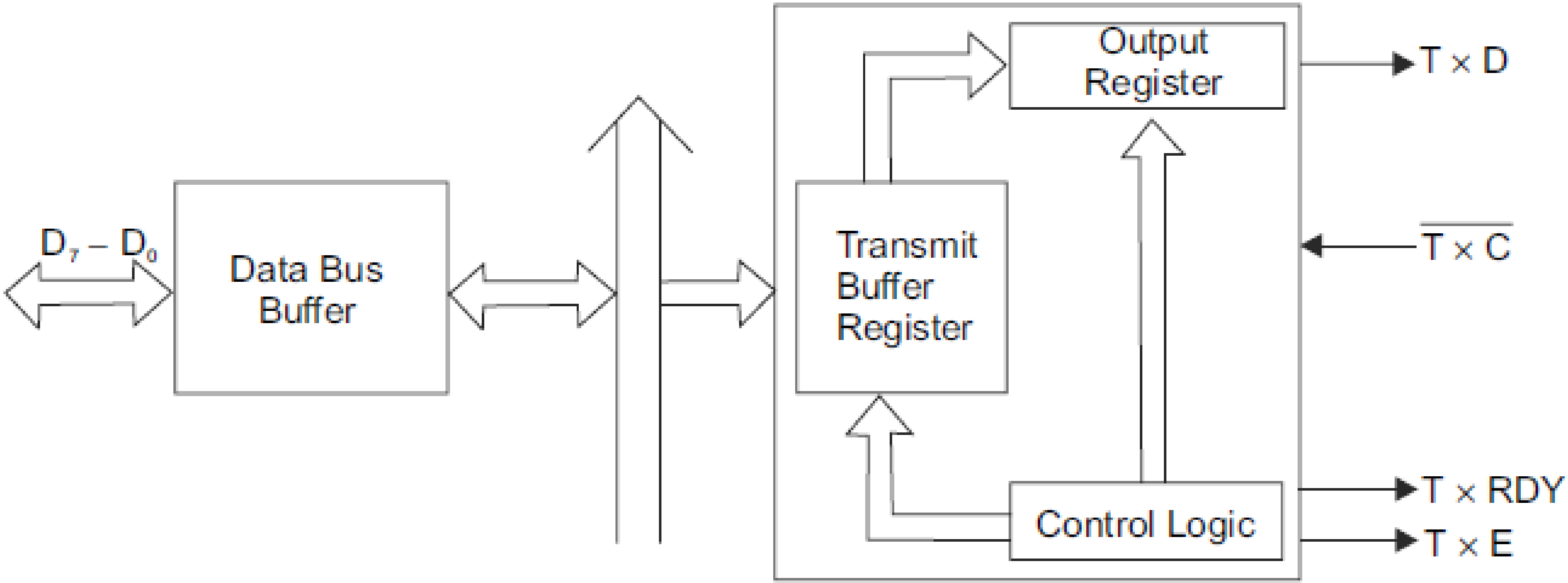
Four Sections:

- Transmitter
- Receiver
- Modem control
- Interface section



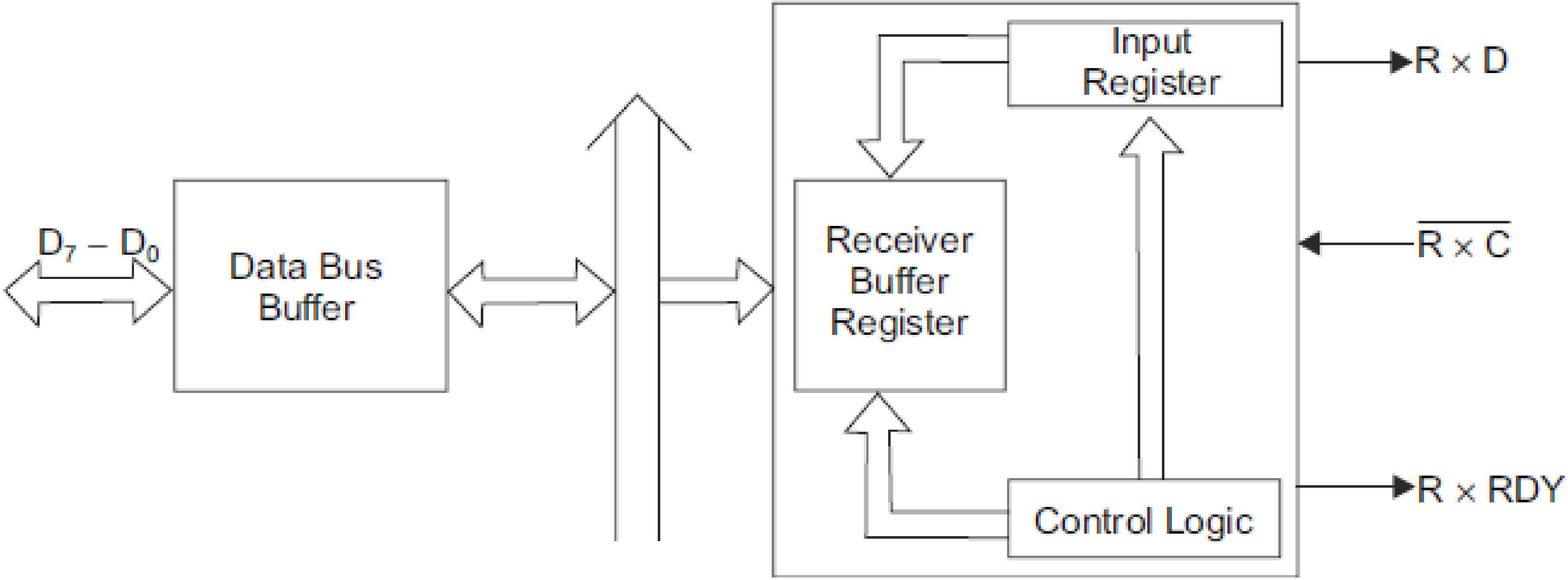
8251 – Functional Block Diagram

Transmitter



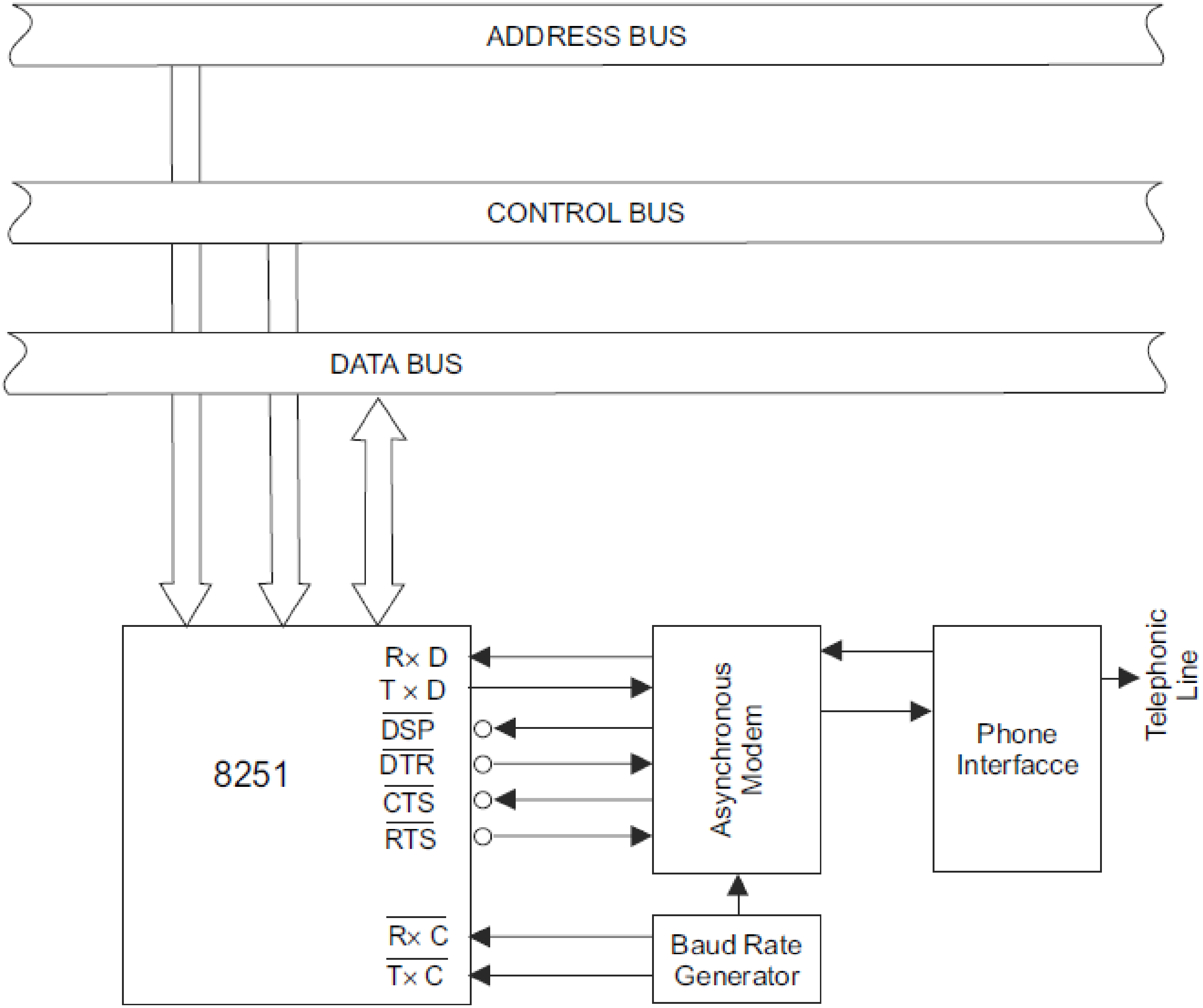
8251 – Functional Block Diagram

Receiver

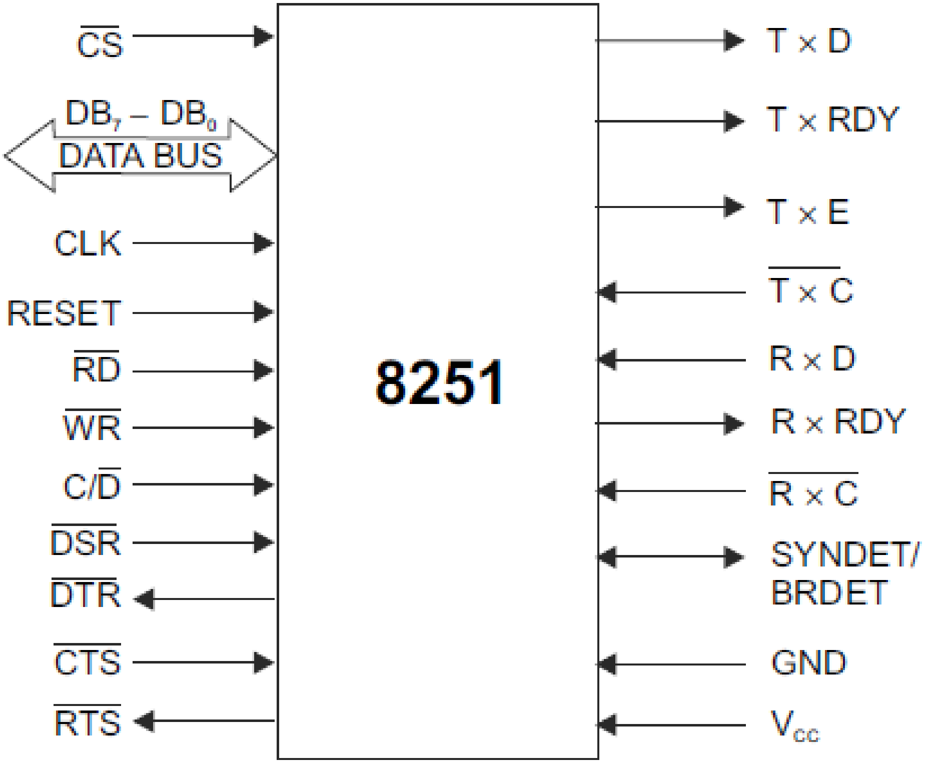


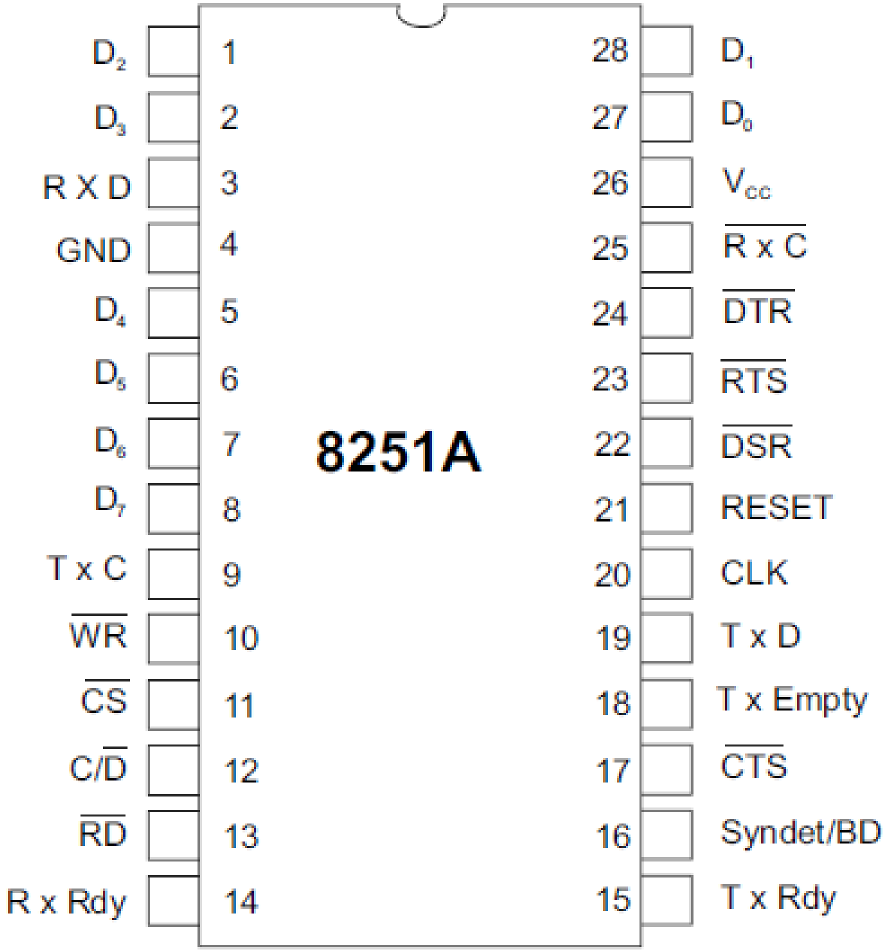
8251 – Functional Block Diagram

Modem Control

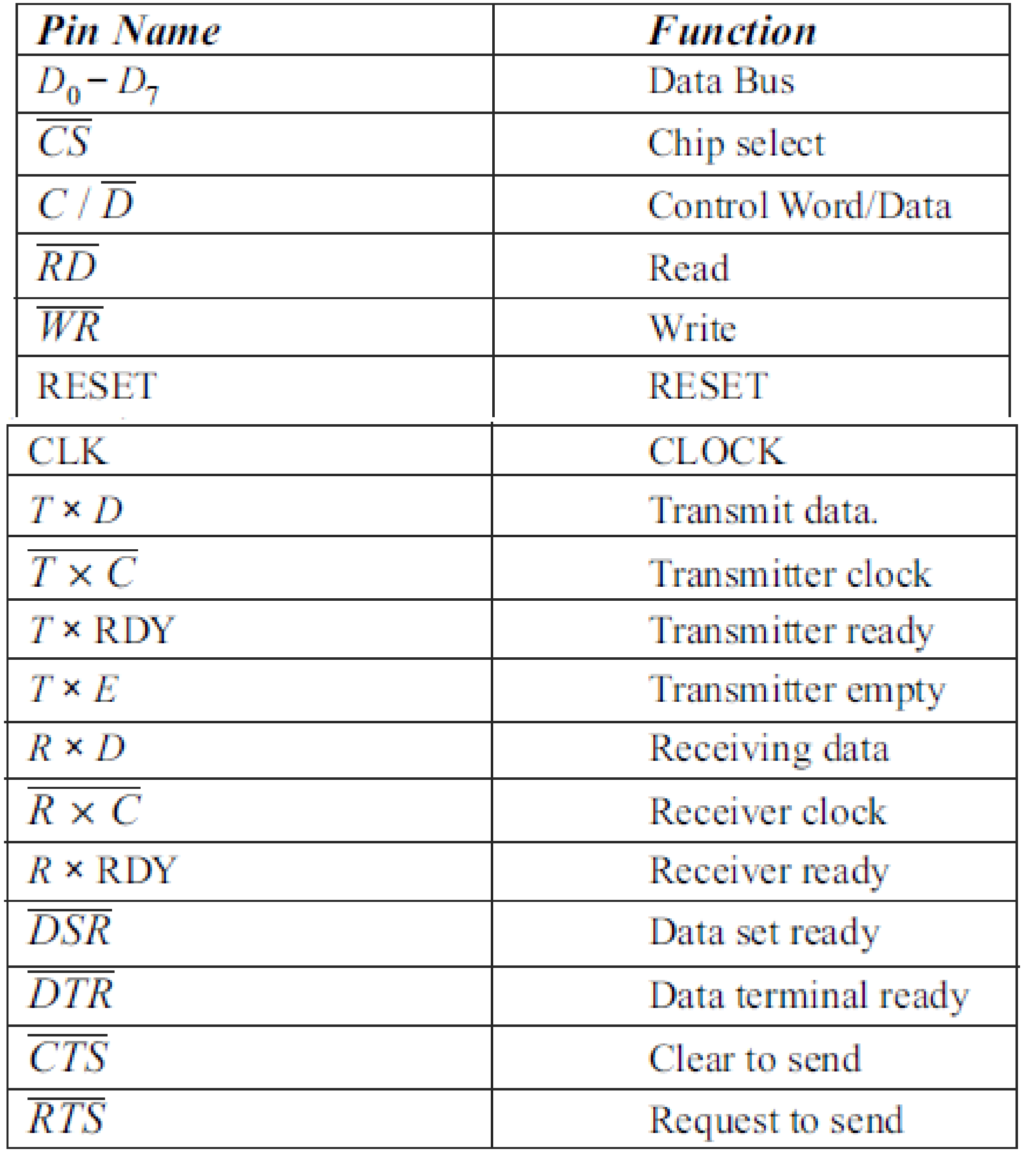


8251 – Pin Diagram





8251 – Pin Diagram



Pin Name	*Function*
$D_0 - D_7$	Data Bus
$\overline{CS}$	Chip select
$C / \overline{D}$	Control Word/Data
$\overline{RD}$	Read
$\overline{WR}$	Write
RESET	RESET
CLK	CLOCK
$T \times D$	Transmit data.
$\overline{T \times C}$	Transmitter clock
$T \times$ RDY	Transmitter ready
$T \times E$	Transmitter empty
$R \times D$	Receiving data
$\overline{R \times C}$	Receiver clock
$R \times$ RDY	Receiver ready
$\overline{DSR}$	Data set ready
$\overline{DTR}$	Data terminal ready
$\overline{CTS}$	Clear to send
$\overline{RTS}$	Request to send

8251 – Pin Diagram

- **D_0-D_7:** The 8-bit data bus is used to read or write status, command word or data from or to the 8251 A.

Read/Write Control Logic:

- $\overline{CS}$ **Chip Select:** An active-low on this input select inputs 8251 A for communication. When $\overline{CS}$ is high, no reading or writing operation can be performed.
- $\overline{\boldsymbol{RD}}$ **Read:** An active-low on this input informs 8251A that the microprocessor is reading either data or status information from internal registers of 8251.
- $\overline{\boldsymbol{WR}}$ **Write:** The active-low input on $\overline{\boldsymbol{WR}}$ is used to inform it that the microprocessor is writing data or control word to 8251.

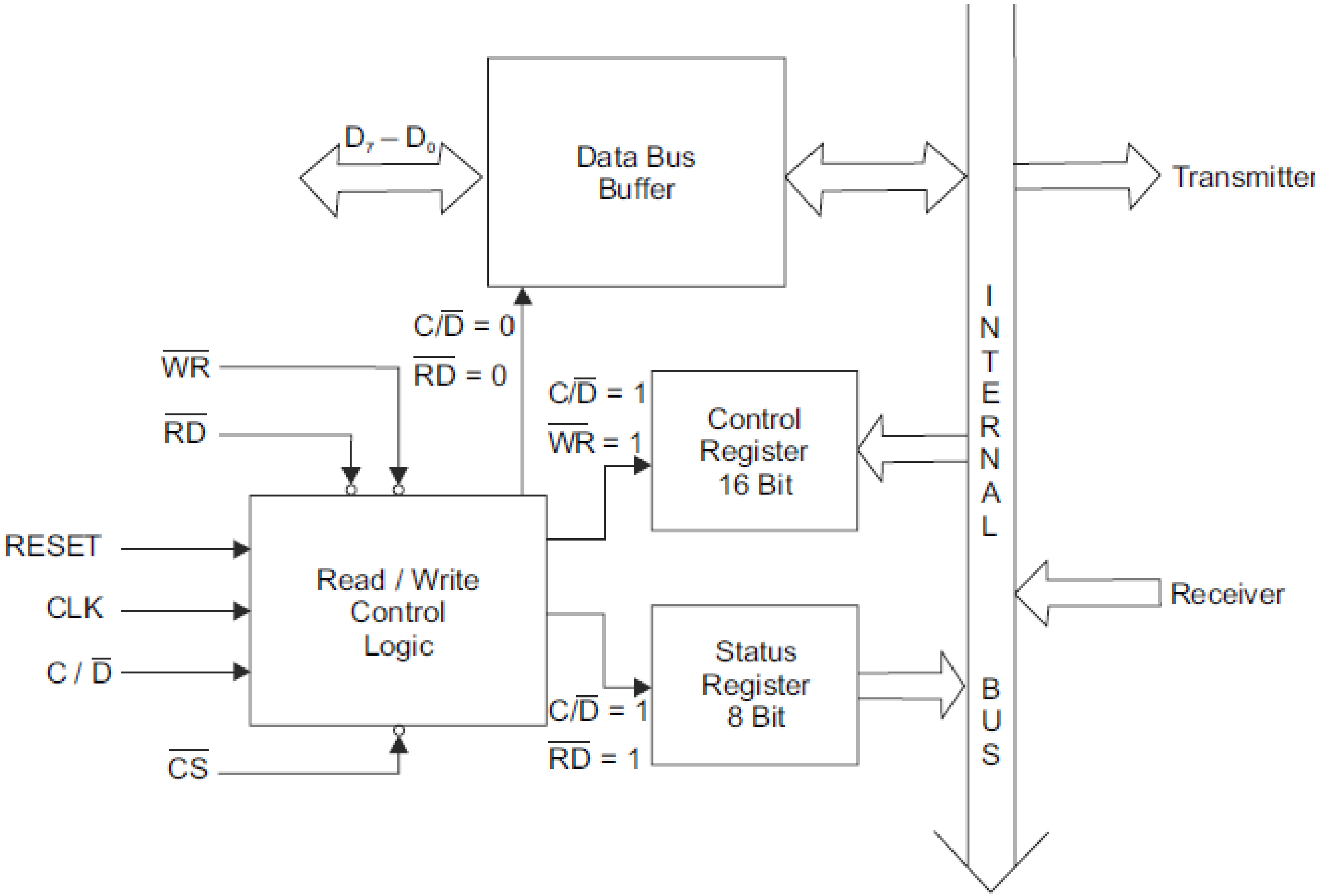
8251 – Pin Diagram

- $C/\overline{D}$ **Control Word/Data:** This pin is used to inform the 8251A that the word on the data bus is either data or control word/status information.

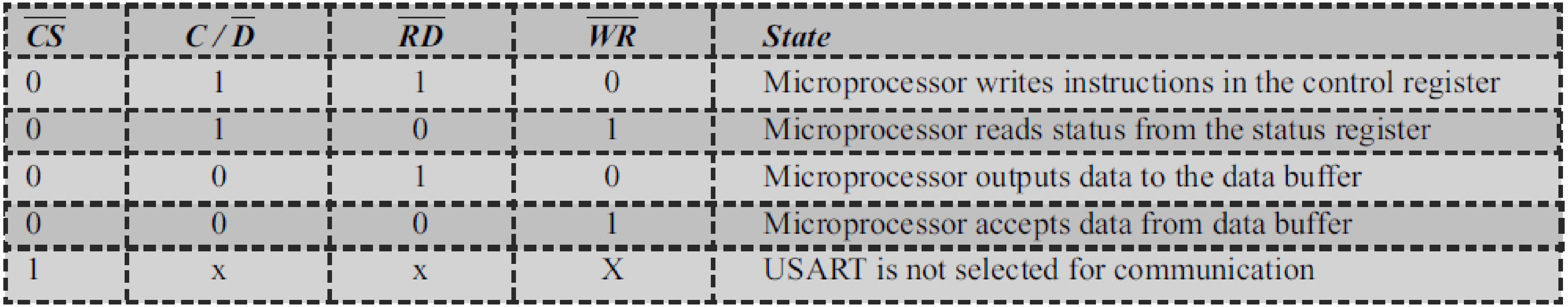
- **RESET:** A high on this input forces the 8251A into an ‘idle’ state. This device will remain idle until a new set of control words is written into it. The minimum required reset pulse width is 6 clock states for each reset operation.

- **CLK:** The CLK input is used to generate internal device timings and is normally connected to the output of a clock generator. The input frequency of CLK should be greater than 30 times the receiver or transmitter data-bit transfer rate.

8251 – Pin Diagram



8251 – Pin Diagram



$\overline{CS}$	$C/\overline{D}$	$\overline{RD}$	$\overline{WR}$	*State*
0	1	1	0	Microprocessor writes instructions in the control register
0	1	0	1	Microprocessor reads status from the status register
0	0	1	0	Microprocessor outputs data to the data buffer
0	0	0	1	Microprocessor accepts data from data buffer
1	x	x	X	USART is not selected for communication

Transmitter:

- **TxD (Transmit Data Output):** The serial data output from the output register is transmitted on T × D pin. The transmitted data bits consist of data along with other informations such as start bit, stop bits and parity bit.
- **TxC (Transmitter Clock Input):** Controls the rate at which the data is to be transmitted. The baud rate is equal to the T × C frequency in synchronous transmission mode. In asynchronous transmission mode, the baud rate is 1, 1/16 or 1/64 times the T×C.

8251 – Pin Diagram

- **TxRDY (Transmitter Ready):** This is output signal, which indicates to the CPU that the transmitter buffer is empty.
- **TxE (Transmitter Empty):** When the T × E output is high, the 8251 has no characters to transmit.

Receiver:

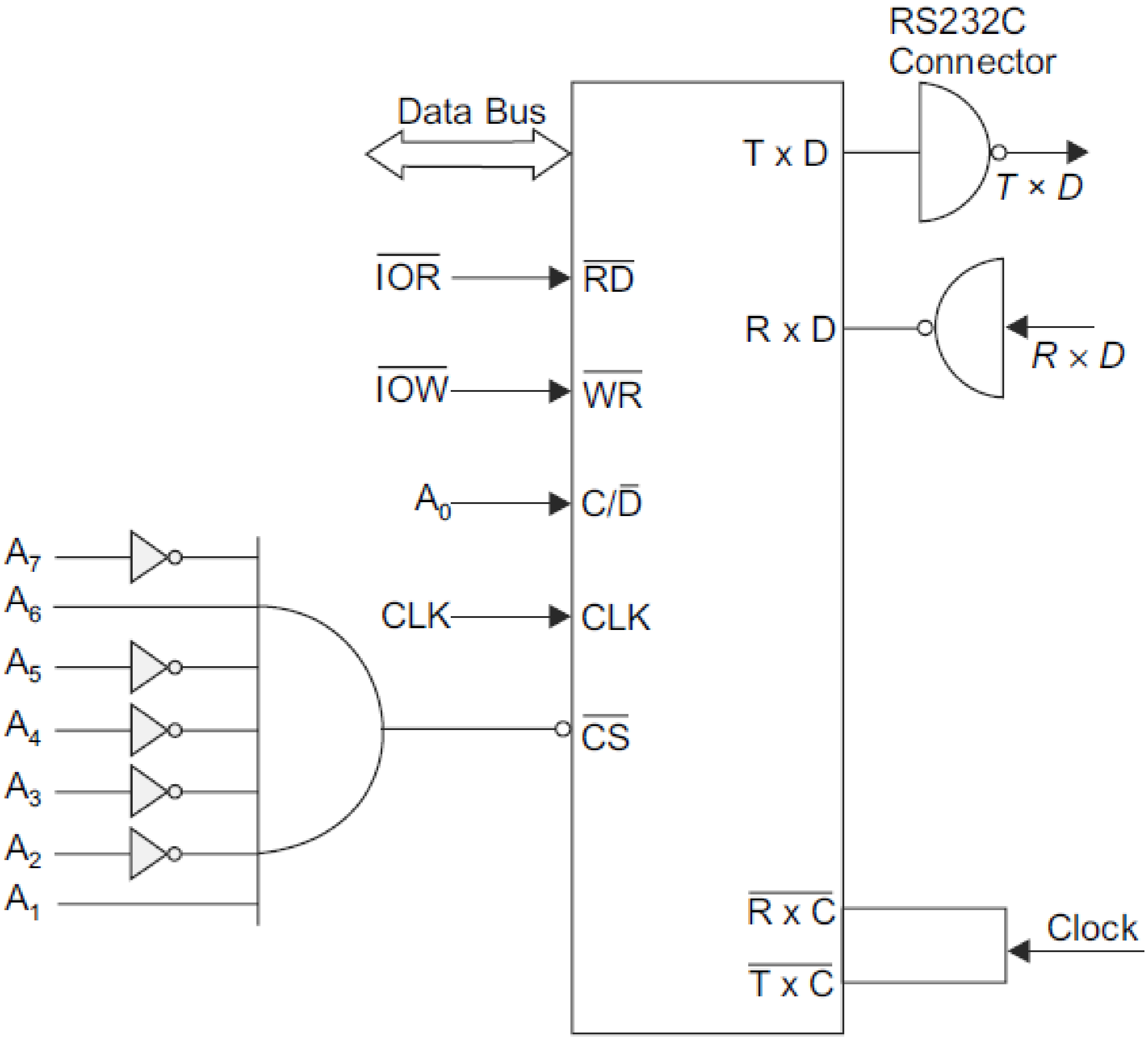
- **RxD Receive Data Input:** This input pin of 8251A receives serial data from outside environment, and delivers to the input register via R × D line which is subsequently put into parallel from and placed in the receiver buffer register.
- **RxC Receiver Clock Input:** The R × C receiver clock input pin controls the rate at which the bits are received by the input register.
- **RxRDY Receiver Ready:** This is an output pin which indicates that 8251A contains a character to be read by the CPU

8251 – Pin Diagram

Modem Control Pins:

- $\overline{\boldsymbol{DSR}}$ **Data Set Ready:** This input can be used as a general-purpose one-bit inverting input port.
- $\overline{\boldsymbol{DTR}}$ **Data Terminal Ready:** This is a general-purpose one-bit inverting output port.
- $\overline{\boldsymbol{CTS}}$ **Clear To Send:** This is a one-bit inverting input port.
- $\overline{\boldsymbol{RTS}}$ **Request To Send:** This is a general-purpose one-bit inverting output port.
- **SYNDET/BD Synchronous Detect / Break Detect:** This pin is used for detection of synchronous characters in synchronous mode and break characters in asynchronous mode.

8251 – Interfacing



8251 – Programming & Operating Modes

- A set of control words can be written into the internal registers of 8251A to make it operate in the desired mode.
- The control words of 8251A are two functional types, namely,
    1. **Mode Instruction Control word**
    2. **Command Instruction Control word**
- There are two 8-bit control registers in 8251 to load the mode word and command word.
- The mode instruction word informs about the initial parameters such as mode, baud rate, stop bits and parity bit.
- The command instruction word explains about enabling the transmitter and receiver section.

8251 – Programming & Operating Modes

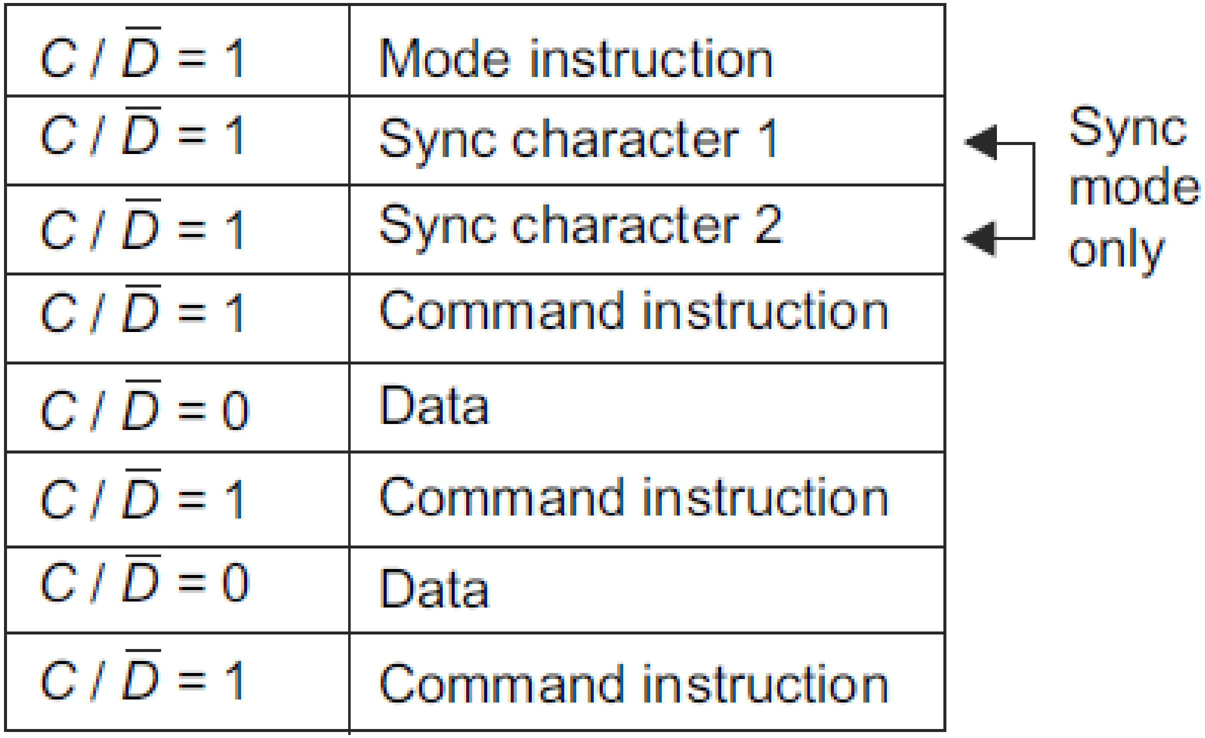
Mode Instruction Control Word:

- Mode instruction control word defines the general operational characteristics of 8251A.
- These control words are different for synchronous and asynchronous mode operation.
- Once the mode instruction control word has been written into 8251, SYNC characters (synchronous mode only) or command instructions (synchronous or asynchronous mode) may be programmed.
- The mode of operation from synchronous to asynchronous or from asynchronous to synchronous can be changed by resetting the 8251.

8251 – Programming & Operating Modes

Mode Instruction Control Word:

Typical Data Block



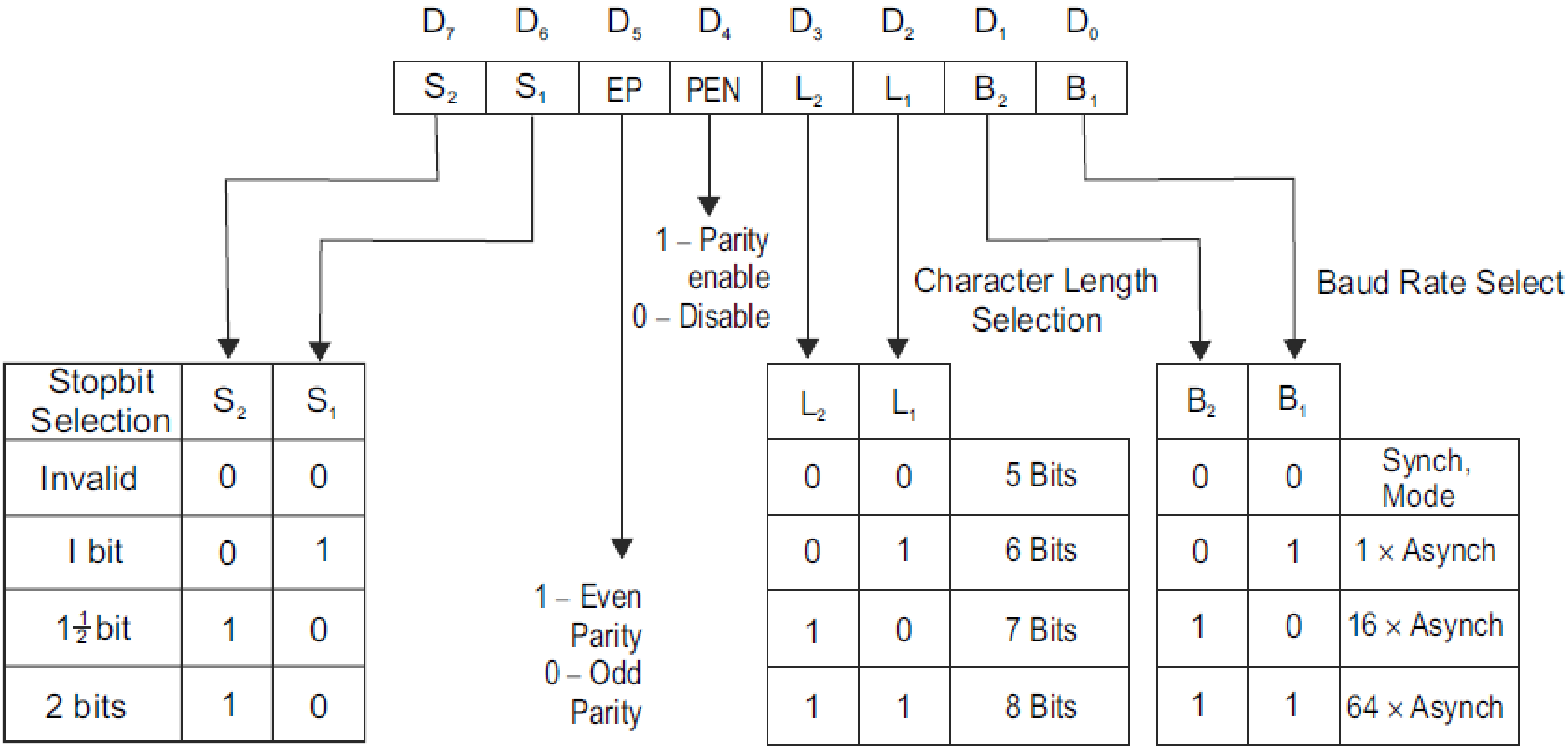
$C/\overline{D} = 1$	Mode instruction
$C/\overline{D} = 1$	Sync character 1
$C/\overline{D} = 1$	Sync character 2
$C/\overline{D} = 1$	Command instruction
$C/\overline{D} = 0$	Data
$C/\overline{D} = 1$	Command instruction
$C/\overline{D} = 0$	Data
$C/\overline{D} = 1$	Command instruction

Sync mode only

8251 – Programming & Operating Modes

Mode Instruction Control Word:

Instruction Format for Asynchronous Mode



D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
S_2	S_1	EP	PEN	L_2	L_1	B_2	B_1

Stopbit Selection	S_2	S_1
Invalid	0	0
I bit	0	1
1½bit	1	0
2 bits	1	0

1 – Even Parity
0 – Odd Parity

1 – Parity enable
0 – Disable

Character Length Selection

L_2	L_1	
0	0	5 Bits
0	1	6 Bits
1	0	7 Bits
1	1	8 Bits

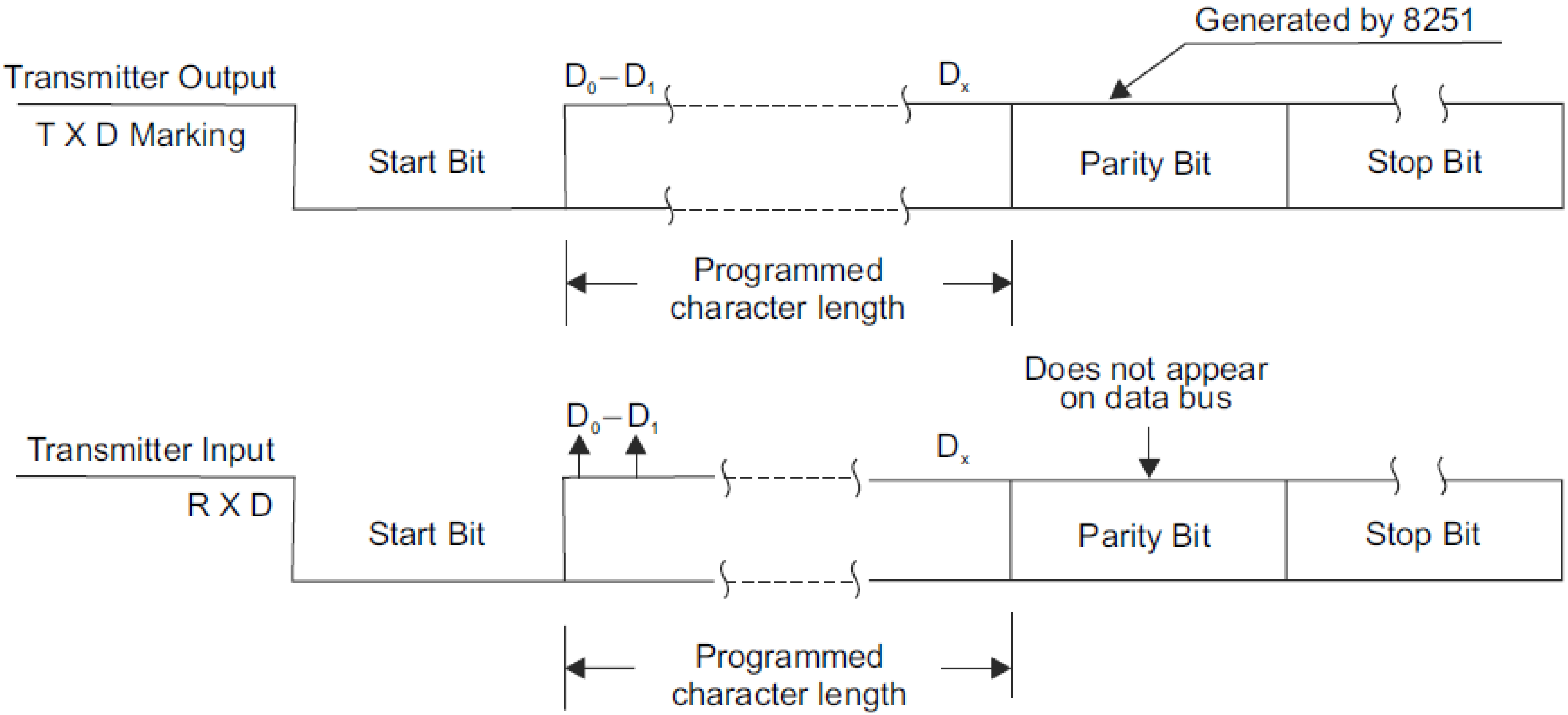
Baud Rate Select

B_2	B_1	
0	0	Synch, Mode
0	1	1 × Asynch
1	0	16 × Asynch
1	1	64 × Asynch

8251 – Programming & Operating Modes

Mode Instruction Control Word:

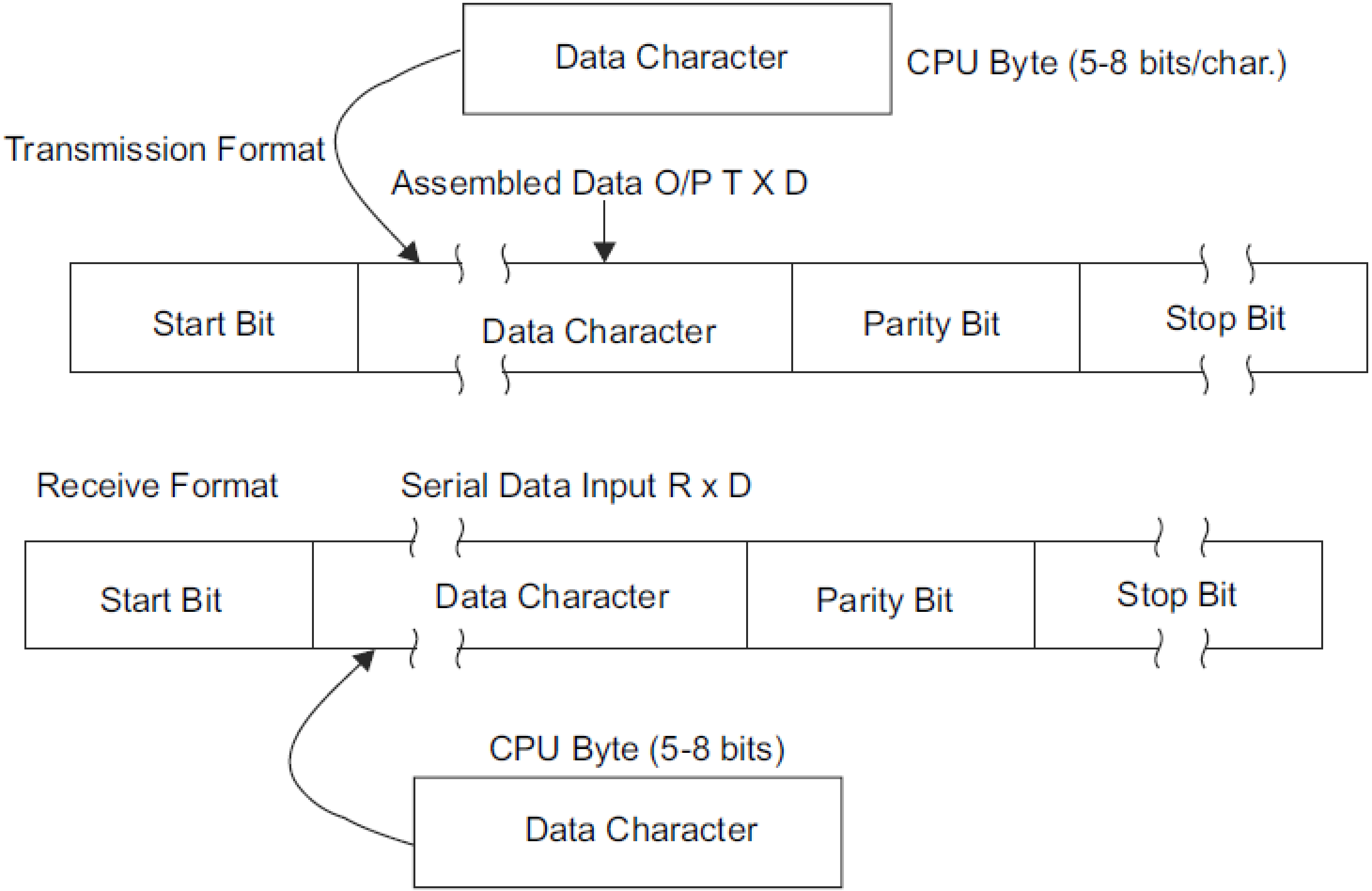
Asynchronous Mode Transmission Format



8251 – Programming & Operating Modes

Mode Instruction Control Word:

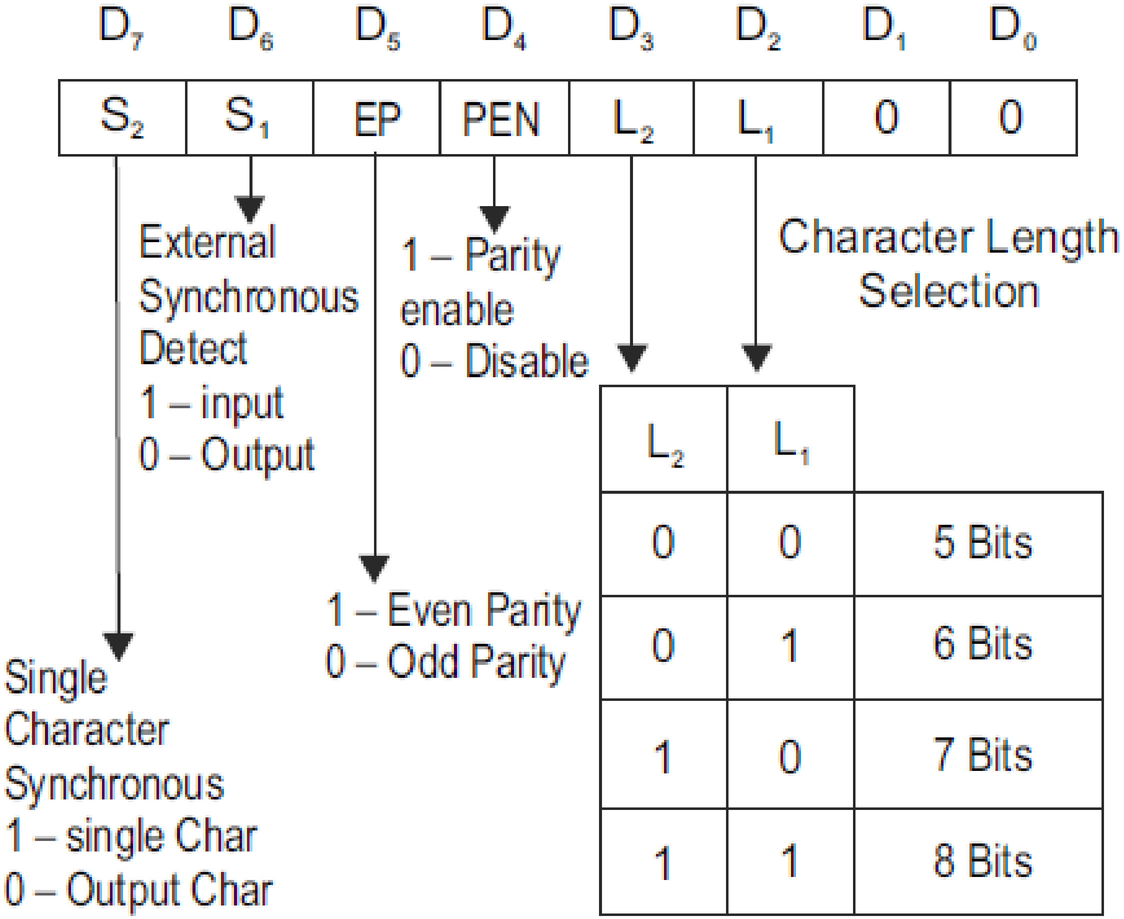
Asynchronous Mode Receive Format



8251 – Programming & Operating Modes

Mode Instruction Control Word:

Instruction Format for Synchronous Mode



D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
S_2	S_1	EP	PEN	L_2	L_1	0	0

Single Character Synchronous
1 – single Char
0 – Output Char

External Synchronous Detect
1 – input
0 – Output

1 – Even Parity
0 – Odd Parity

1 – Parity enable
0 – Disable

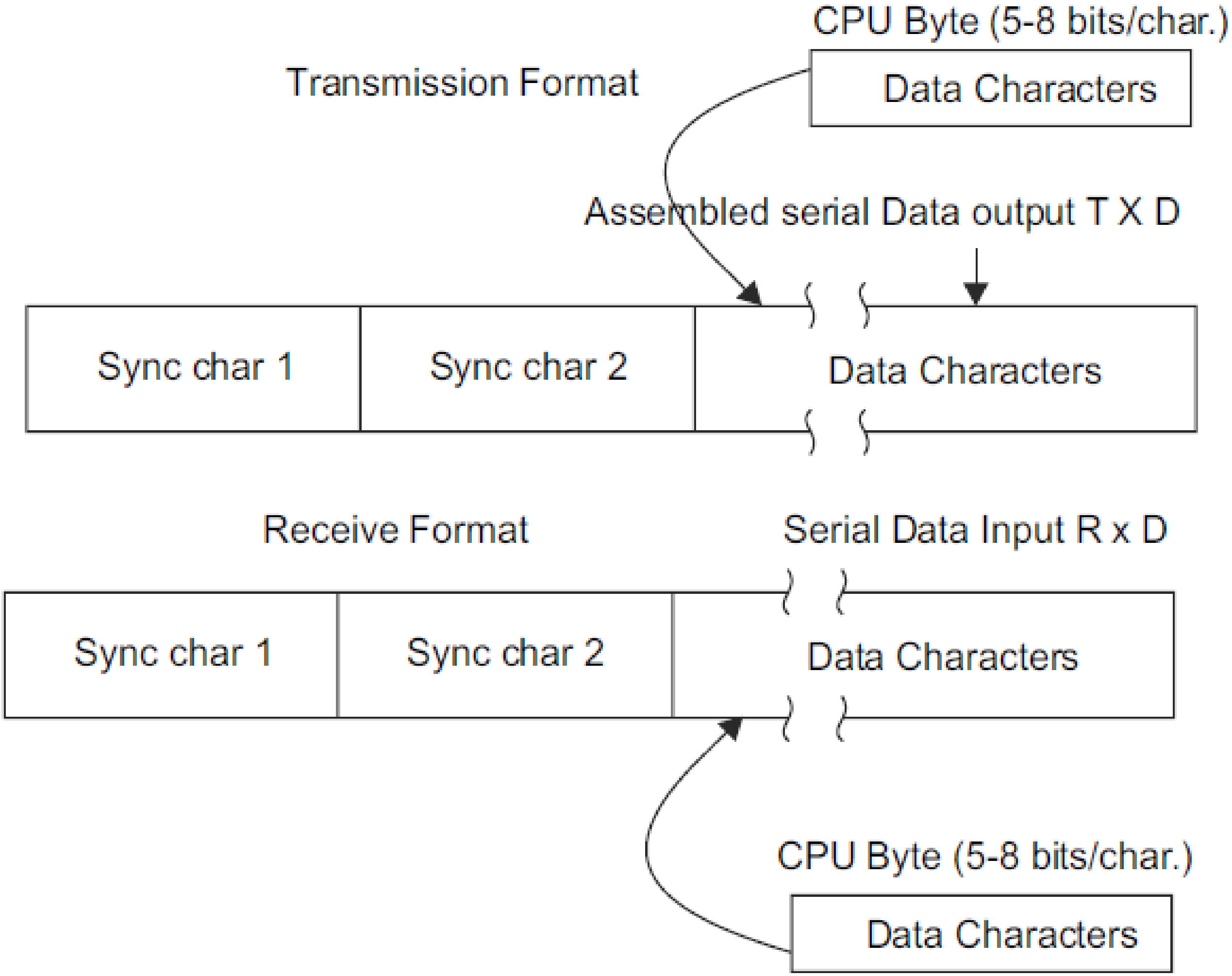
Character Length Selection

L_2	L_1	
0	0	5 Bits
0	1	6 Bits
1	0	7 Bits
1	1	8 Bits

8251 – Programming & Operating Modes

Mode Instruction Control Word:

Synchronous Mode Transmission & Receive Format



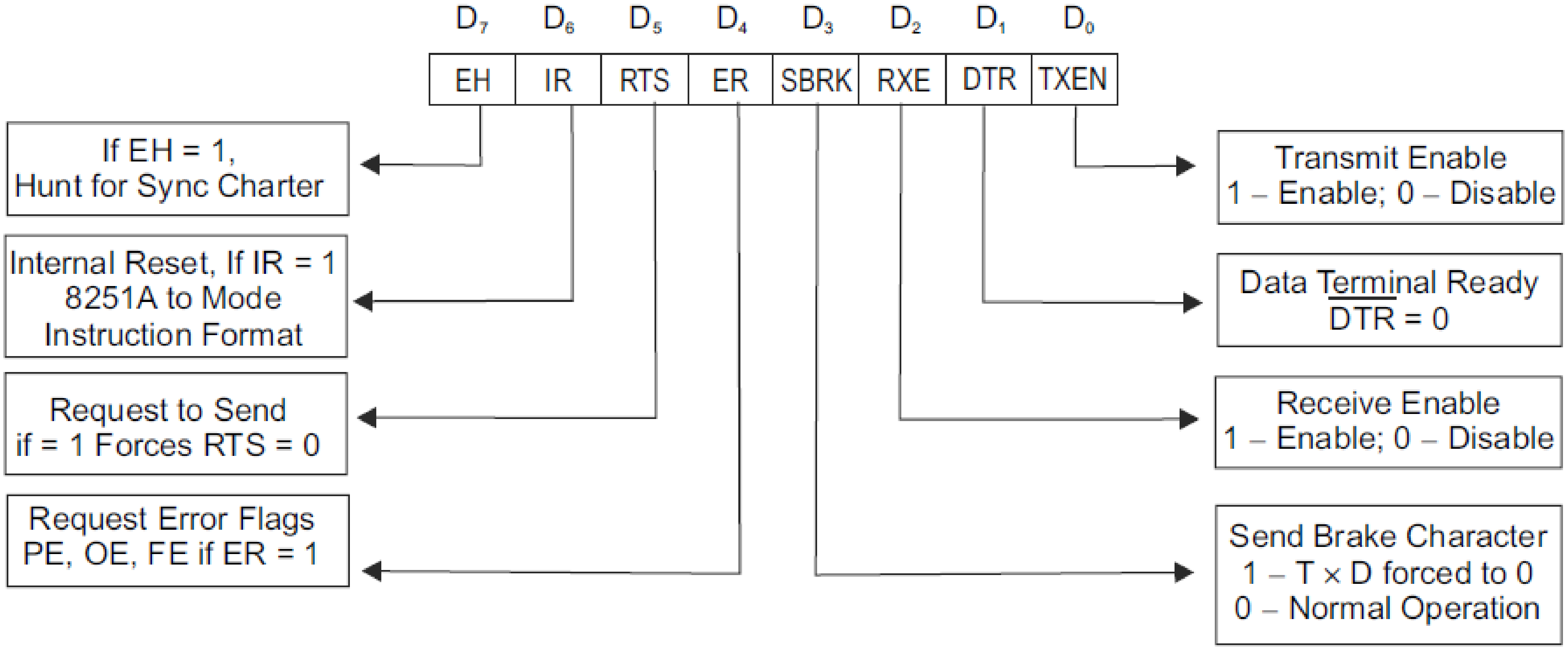
8251 – Programming & Operating Modes

Command Instruction Word:

- The command instruction controls the actual operations of the selected format like enable transmit/receive, error reset and modem controls.
- Once the mode instruction has been written into 8251A and the SYNC characters are loaded (only in synchronous mode), the device is ready for data communication.
- The command instructions can be accepted only after mode instruction in case of asynchronous mode.
- All further control words written with C/$\overline{D}$ will load a command instruction.
- A reset operation returns the 8251 back to mode instruction format from the command instruction format.

8251 – Programming & Operating Modes

Command Instruction Word:



D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
EH	IR	RTS	ER	SBRK	RXE	DTR	TXEN

If EH = 1,
Hunt for Sync Charter

Internal Reset, If IR = 1
8251A to Mode
Instruction Format

Request to Send
if = 1 Forces RTS = 0

Request Error Flags
PE, OE, FE if ER = 1

Transmit Enable
1 – Enable; 0 – Disable

Data Terminal Ready
$\overline{\text{DTR}} = 0$

Receive Enable
1 – Enable; 0 – Disable

Send Brake Character
1 – T × D forced to 0
0 – Normal Operation

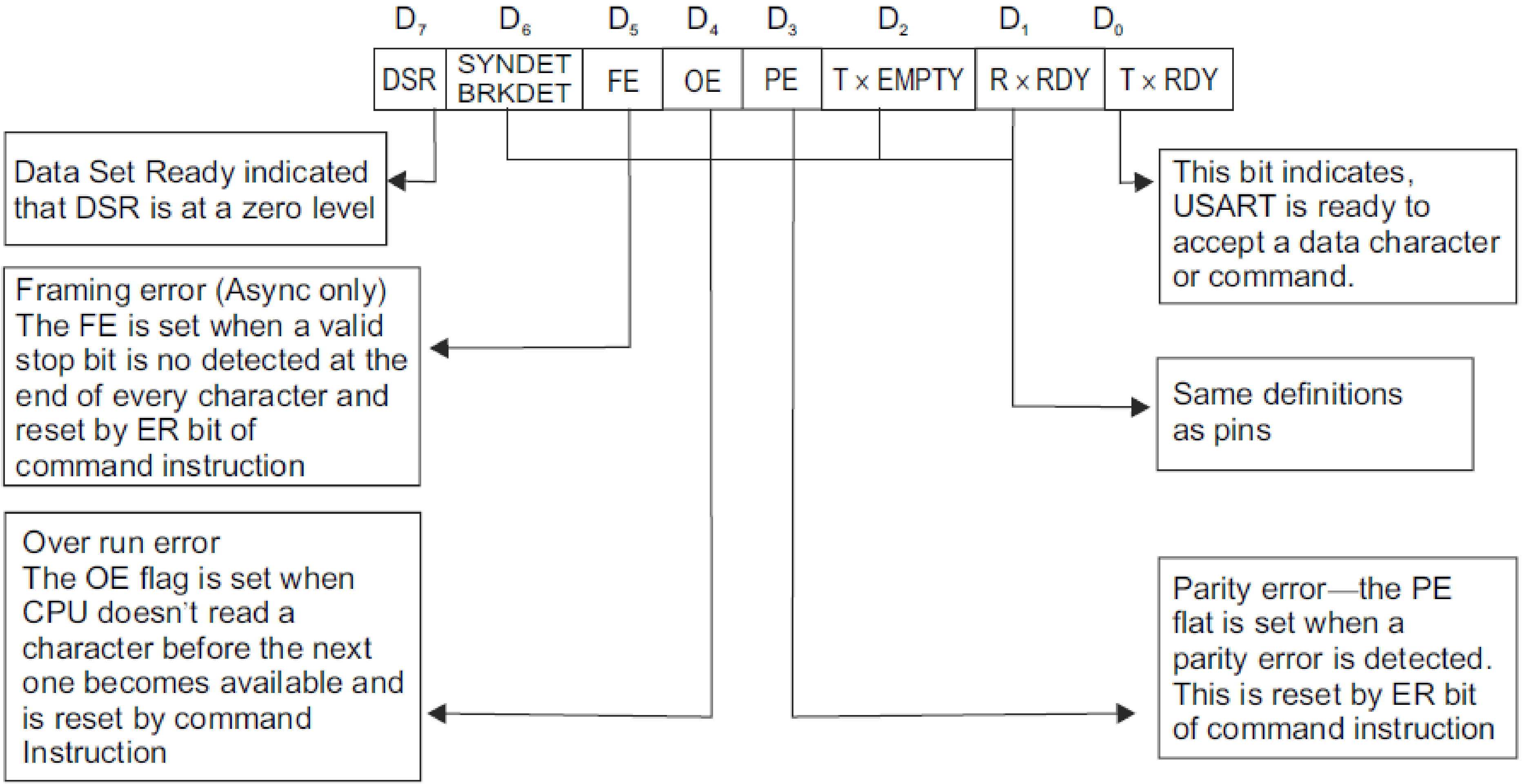
8251 – Programming & Operating Modes

Status Word Register Format:

- The status word can be read with C/$\overline{D}$ =1.
- The CPU requires various information to operate properly.
- All required information are provided by the status word.
- The status word is continuously updated by 8251, except when CPU reads the status word.

8251 – Programming & Operating Modes

Status Word Register Format:



D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
DSR	SYNDET BRKDET	FE	OE	PE	T x EMPTY	R x RDY	T x RDY

Data Set Ready indicated that DSR is at a zero level

Framing error (Async only) The FE is set when a valid stop bit is no detected at the end of every character and reset by ER bit of command instruction

Over run error The OE flag is set when CPU doesn't read a character before the next one becomes available and is reset by command Instruction

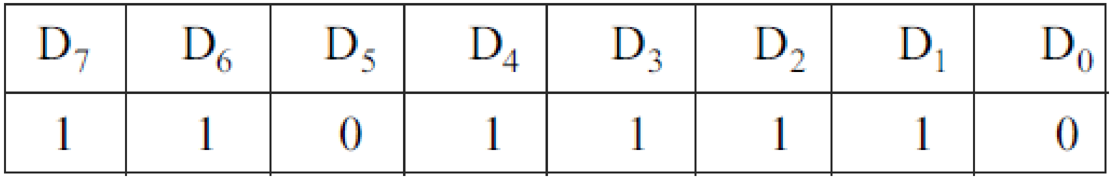
This bit indicates, USART is ready to accept a data character or command.

Same definitions as pins

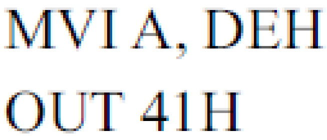
Parity error—the PE flat is set when a parity error is detected. This is reset by ER bit of command instruction

8251 – Programming & Operating Modes

Problem 1: Find the mode instruction for the following operations: 8251 can be operated in asynchronous mode for data transmit, The baud rate is 16 × Asynch, The length of character is 8 bits and number of stop bits is 2. Assume odd parity, the address of the control register is 41H and the address of data register is 40H.



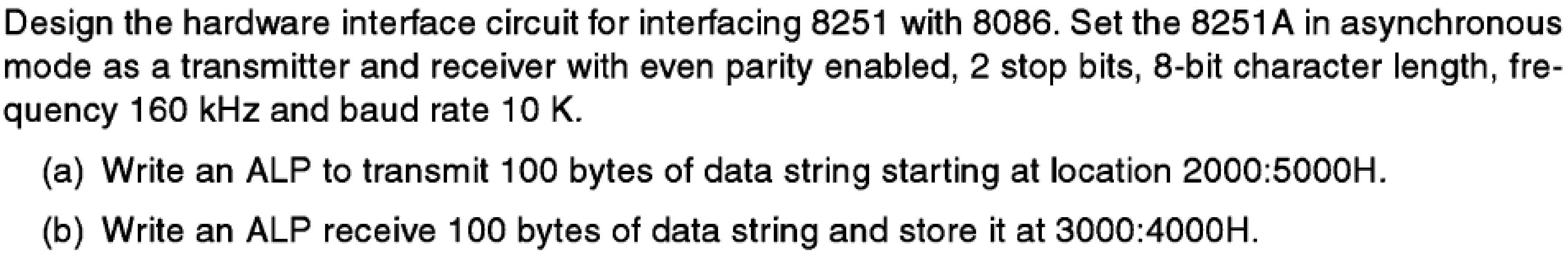
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	1	0	1	1	1	1	0



```
MVI A, DEH
OUT 41H
```

8251 – Programming & Operating Modes

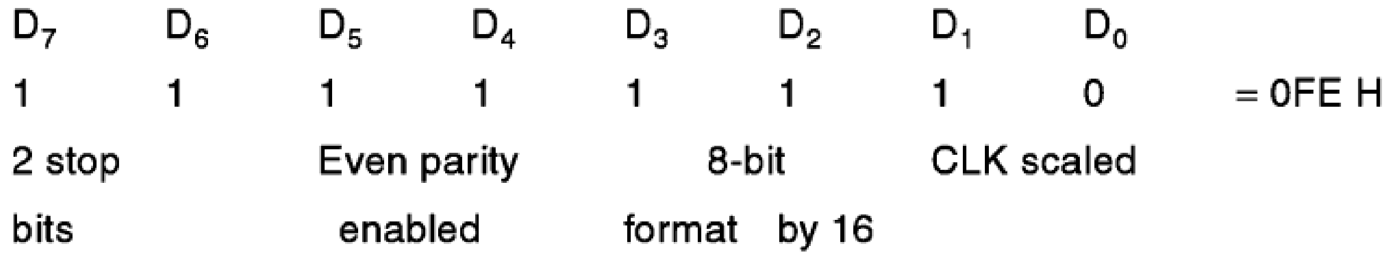
Problem 2:



Design the hardware interface circuit for interfacing 8251 with 8086. Set the 8251A in asynchronous mode as a transmitter and receiver with even parity enabled, 2 stop bits, 8-bit character length, frequency 160 kHz and baud rate 10 K.

(a) Write an ALP to transmit 100 bytes of data string starting at location 2000:5000H.

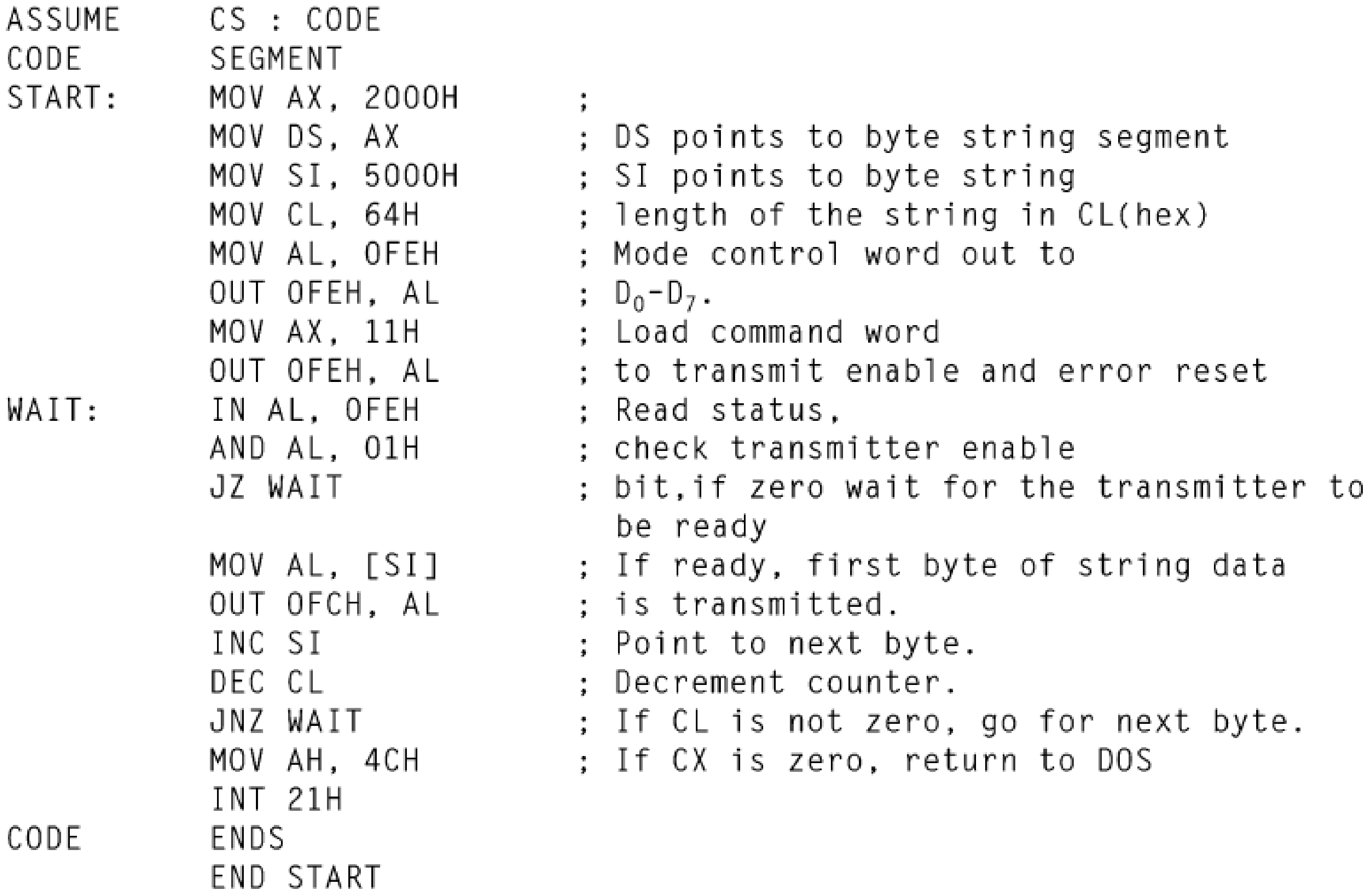
(b) Write an ALP receive 100 bytes of data string and store it at 3000:4000H.



D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
1	1	1	1	1	1	1	0	= 0FE H
2 stop bits		Even parity enabled		8-bit format		CLK scaled by 16		

8251 – Programming & Operating Modes

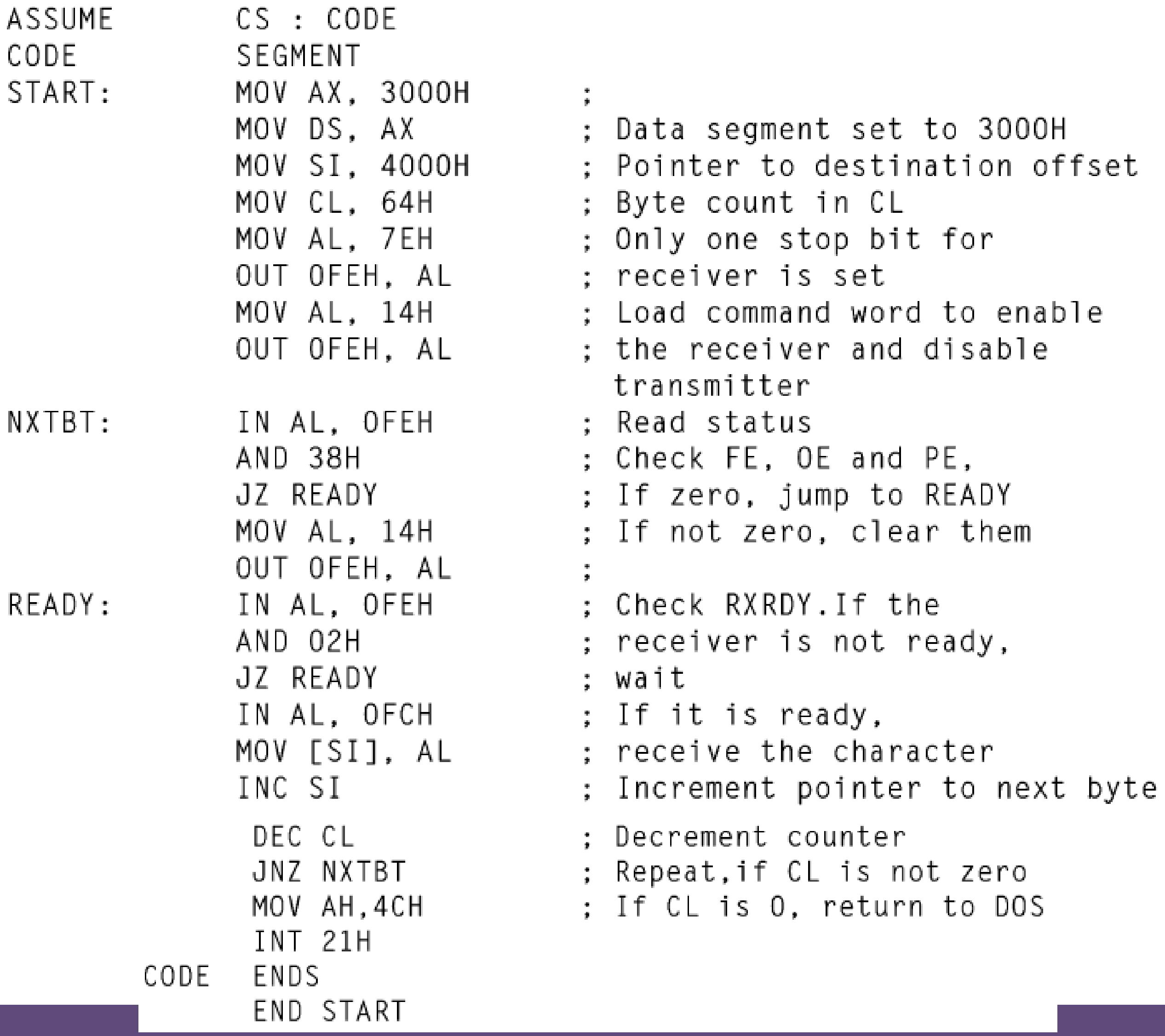
Problem 2: ALP to Transmit 100 bytes



```
ASSUME    CS : CODE
CODE      SEGMENT
START:    MOV AX, 2000H        ;
          MOV DS, AX           ; DS points to byte string segment
          MOV SI, 5000H        ; SI points to byte string
          MOV CL, 64H          ; length of the string in CL(hex)
          MOV AL, 0FEH         ; Mode control word out to
          OUT 0FEH, AL         ; D0-D7.
          MOV AX, 11H          ; Load command word
          OUT 0FEH, AL         ; to transmit enable and error reset
WAIT:     IN AL, 0FEH          ; Read status.
          AND AL, 01H          ; check transmitter enable
          JZ WAIT              ; bit.if zero wait for the transmitter to
                                 be ready
          MOV AL, [SI]         ; If ready, first byte of string data
          OUT 0FCH, AL         ; is transmitted.
          INC SI               ; Point to next byte.
          DEC CL               ; Decrement counter.
          JNZ WAIT             ; If CL is not zero, go for next byte.
          MOV AH, 4CH          ; If CX is zero, return to DOS
          INT 21H
CODE      ENDS
          END START
```

8251 – Programming & Operating Modes

Problem 2: ALP to Receive 100 bytes



```
ASSUME      CS : CODE
CODE        SEGMENT
START:      MOV AX, 3000H      ;
            MOV DS, AX         ; Data segment set to 3000H
            MOV SI, 4000H      ; Pointer to destination offset
            MOV CL, 64H        ; Byte count in CL
            MOV AL, 7EH        ; Only one stop bit for
            OUT 0FEH, AL       ; receiver is set
            MOV AL, 14H        ; Load command word to enable
            OUT 0FEH, AL       ; the receiver and disable
                                 transmitter
NXTBT:      IN AL, 0FEH        ; Read status
            AND 38H            ; Check FE, OE and PE,
            JZ READY           ; If zero, jump to READY
            MOV AL, 14H        ; If not zero, clear them
            OUT 0FEH, AL       ;
READY:      IN AL, 0FEH        ; Check RXRDY.If the
            AND 02H            ; receiver is not ready,
            JZ READY           ; wait
            IN AL, 0FCH        ; If it is ready,
            MOV [SI], AL       ; receive the character
            INC SI             ; Increment pointer to next byte

             DEC CL            ; Decrement counter
             JNZ NXTBT         ; Repeat,if CL is not zero
             MOV AH,4CH        ; If CL is 0, return to DOS
             INT 21H
       CODE  ENDS
             END START
```