Digital Systems Lab Assignment - 2

Assignment by: Pranav Sutar
CS20B029

- Objectives :

1) Understanding how a character is interpreted by computer systems.

2) Converting an ASCII code into BCD code.

3) Converting a BCD code to ASCII code.

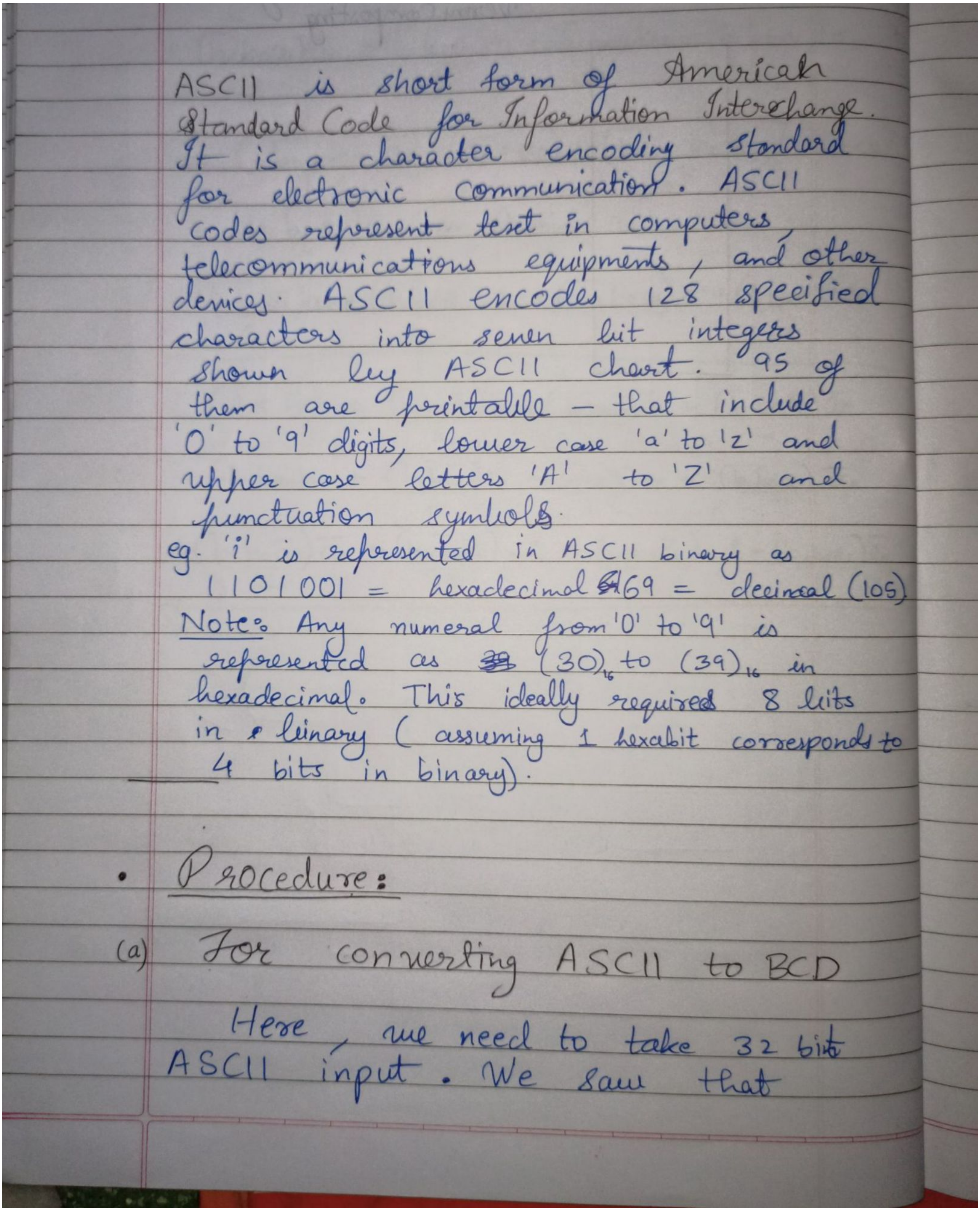
4) To learn to write test bence.

5) To ~~ched~~ check whether a character is numeral ('0' to '9') or not.

6) To concatenate or append variables in Verilog HDL.

- Description:

BCD, is binary-coded decimal. BCD is is a class of binary encodings of decimal numbers where each digit is represented by a fixed, (usually 4) number of bits. BCD's main virtue is its more accurate representation, and rounding of decimal quantities, as well as conversion into conventional human readable interpretations



ASCII is short form of American Standard Code for Information Interchange. It is a character encoding standard for electronic communication. ASCII codes represent text in computers, telecommunications equipments, and other devices. ASCII encodes 128 specified characters into seven bit integers shown by ASCII chart. 95 of them are printable – that include '0' to '9' digits, lower case 'a' to 'z' and upper case letters 'A' to 'Z' and punctuation symbols.

eg. 'i' is represented in ASCII binary as 1101001 = hexadecimal 69 = decimal (105)

Note: Any numeral from '0' to '9' is represented as $(30)_{16}$ to $(39)_{16}$ in hexadecimal. This ideally requires 8 bits in binary (assuming 1 hexabit corresponds to 4 bits in binary).

- Procedure:

(a) For converting ASCII to BCD

Here, we need to take 32 bit ASCII input. We saw that ideally a character '0' to '9' is represented by 2 hexabits from $(30)_{16}$ to $(39)_{16}$ which at most requires 8 bits. Thus 32 bit ASCII comprises of 4 numeral charaters. To obtain the BCD code of it, we must divide the 32 bits into four equal parts of 8 bits The first half of ~~ech~~ each group is (0011) which corresponds to $(3)_{16}$. If thats not the case, then output is not valid.

We ~~z~~ just need to concatenate the right halfs of each ~~g~~ of the 4 group without changing their relative positions.

eg. Consider 32h' 34373231

Here, dividing into four parts, we get 34 | 37| 32 | 31. as the first part of each is 3, hence it is valid. ~~So~~ we just need to cancatenate the ~~sech~~ second half i.e. (4 7 21)~~16~~ decimal

≡ (0100 0111 0010 0001) BCD

In this way, we can convert 32-bit binary to 16-bit ASCII.

(b) BCD (at most 16 bits) to ASCII.

This is just the opposite process of the previous process, and hence very easy. But here, there is no need to verify whether the input is valid or not.

We need to divide inputs into group of 4 bits from LSB. at each breakpoint and ~~after the~~ to the rightmost part, we insert (0011), which ~~a~~ automatically converts it into 32 lit ASCII code.

eg. $(9814)_{10}$ is the ~~ascii input~~ BCD input
i.e. (1001 1000 00010100). is
(0011 1001 0011 1000 0011 0001 0011 0100)$_2$ → ASCII format.

Result: (waveforms are in zip file)

(a) ~~(i)~~ ASCII to BCD

(i) When ASCII = 32'h 37 37 33 36,
valid = 1 and
BCD = 0111 0111 0011 0110.

(ii) when ASCII = 32'h 31 39 38 37
BCD = 0001 1001 1000 0111, valid = 1

(b) BCD to ASCII

(i) When BCD = 16'h 3272,
ASCII = 32'h 33 32 37 32

(ii) when BCD = 16'd 50876,
ASCII = ~~0011 32'h~~
= 0011 110000 011011000 111011 00111100.