Version 1.0

Section 0: Pre-Lab

• Matrix Multiplication depends on how matrix columns and rows match. We need a 1x4 nibble times a 4x8 generator to generate a suitable code. The cols and rows MUST match, else the bounds will not work.

	O Pre	-lab +H		
O O O O I I O O O O O				
0 0 0 1 1 0 0 0 0 0	1,)	6=/1000	0111	
0 0 0 1 0 1 0 0 0 0 0 0 0				
0 0 0 1 1 0 0000: (0.1)+(0.0)+(0.0)+(0.0) = 0 efc for each colomn. It is easy to check how many units the answer will be become a 1xy yx8 will be a 1x8. M. Nector form Binary form Bue 16. 0000 00000000 0000 0000 0000 0000 00				
0000: (0.1)+(0.0)+(0.0)+(0.0) = 0 efc for each colimn. It is easy to theck how many units the answer will be because a 1xy yx8 will be a 1x8. M. Nector form Binary form Bove 16. 0000 00000000 0000 0000 0000 0000 00				
Efc for each colimn. It's easy to check how many unit the answer will be become a 1'xy yxs will be a 1xs. M. Nector firm. Binary form Bue 16. 0000 00000000 000 000 000 000 000 000				
Efc. for each column. It is easy to check how many units the answer will be because a 12xy 1 yx8 will be a 1x8. M. Nector figm. Binary form Bove 16. 0000 00000000 0000 0000 0000 0000 00	000	0: (0.1)+(0.0)+	10.01 +10.01 =0	
THI easy to Check how many units the answer will be because a 11x4 . 4x8 will be a 1x8.		eft. for each	/pliumin " //	
Will be becase a 1xy 1 4x8 will be a 1x8. W Vector form Binary form Box 16 0000 0000 0000 0000 00000000 000000	(·I	tis easy to thech	how many mits the	unswer
M Vector fam Binery form Bue 16 0000 0000 0000 0000 00000000 000000	wil	The belove a	1'x4 + Ux8 will be	a 1x8.
M Vector farm Binary form Bue 16 0000 0000 0000 0000 0000 0000 0000				n the
0000 00000000 0000 0000 0000 0000 0000 0000	m	Nector form		Bare 16
0001 00010001 1000100 0 0x11 0010 010 0 0x22 0011 001100 0100 010 0 0x33 0100 01000100 0010 0010 0 0x33 0100 01000100 0010 0010 0 0x35 0100 01001001 001001001 001001001 00x55 0100 01001001 00x00 00x0010001 0x88 001 001001 00010001 0x88 001 001001 00x00 00x001 0x88 001 001001 0x99 0100 01001 0x88 0100 01001 0x66 0100 01001 0x66 0x6	000			
0010 001000 0 0100010 0 0x22 0011 0011001 1100 1000 0x33 0100 01000100 0010001 00x33 0100 01000100 0010001 00x55 010 0110011 010001	0001	00010001	10001000	0 x 11
001 0011001 1 1100 1100 0x33 0100 01000100 0010 0010 0010 0x44 0101 0101010 0100 0010 001 000 0x55 010 01100110 0100 010001 0x66 011 0111011 110 110 0010001 0x88 1001 1001 1001 0010 010001 0x88 1001 1001 1010 01010 01010 0x88 1100 1100 100 001 001 0x88 1100 1100 100 001 001 0x88 1100 1100 100 001 001 0x88	0010	0000000	01000100	
0100 01000100 0010 0010 0010 0010 0055 010 0100110 0100 010	0011			0x33
0 10 0 11 0 0 1 0 0 1 0 0 0 1 0 0 0 0 0	0 00	01000100		0×44
0 0 1 0 1 0 0 0 0 0	0101	01010101	10101010	0×55
1000 1000 1000 0001 000	0110	01100110	0110 0110	0×66
100	0111	01110111	1110 1110	OX77
100 1010 1010 01010 0	1000	to a fill a service of the fill and the service of	00010001	UX88
10	1001	10011001	1001 1001	0×99
1100 1100 100 001 001 0xcc 1101 1101 1011 0x00 0x00 0x66	1010			OXAA
110 1101 1101 1011 1011 0XDD			11011101	OXBB
1110 1110 1110 0111 0111 0x 66			The state of the s	OXCC
	Annual Contract		1011 1011	OXDD
OXFE	1110			
	1101	1111111	11111111	OXFF
9				
	9			1

-	
20	a.) 11100011 Hr
	11100011
	(1)
	(1000111)(HT)
	(1 T, 0 to to) = 1
	(+1) %2 /= 2
	(++++)
	(1+1+1) = 10 0 0 0 1
	e= (1011)
	Syndiame vector says, 2nd row of Harm
	and many prompted to the of the
	0 - 11000 701111
	Since the data bits don't match parity
	there might have been 2 errors
	AT COMPRESENT LENGTS
	11.0 - 11.0 - 9110
	(1,1) - 2
	CIT IN-10 IN DESIGNATION
	(1) =1 (0101)
	(++) > 0
	101101101101101
	Ham Err, More than lerror, it points
	to an invalid Hem in HT. Connot be
	Corrected.
	Coversed.

-	3) Louis table [16].
	0.0
	1; 4
	2.5
	3.) err
	4.) 6
	5.) err
	63 err
	7.1 3
	8.) 7
	g) err
	10) err
	11) 2
	12.1 err
	13.) 1
	14) 0
	17.) 0.
9	15.) err
	The state of the s

Section 1: Bm.c

- Since there will be a lot of segmenting into the correct byte for this ADT, I created a function based off Euegene's idea in his section that takes in a certain number of bits and returns the correct number of bytes. This function in particular will be very useful for selecting the correct byte for a row.
- For bm_create, make sure to allocate the correct amount of memory for each row, not the number of cols rather the number of bytes necessary to fit the number of cols. 0-8 cols should correspond to 1 byte allocated.
- For set/clr/get bit apply binary manipulation. Create a manipulator byte, 0x01 which can be used to access a byte and return a value at a certain index
- For bm_print, pretty simple just get the bit at the iterator i,j combo.

Section 2: Hamming.c

- For practice purposes I'm going to create temporary ADT's G and H to test my code. Also add professors upper and lower nibble code, which is extremely handy for byte manipulation.
- Ham init will create a 4x8 bitmap G and a 8x4 bitmap H

- To create G, iterate through the number of cols then 4 times which can be used for the generator matrix's 4 rows and the 4 bits of data. H is the same but swap some things.
- To encode the data,
 - First we get the lower nibble of the data byte
 - O Create 2 variables: 1 to hold the result which will be or'd with the other variable which holds the value of the current matrix vector row/col multiplication.
- To decode the generated 8 bit hamming code
 - Multiply the inputed code by H transpose to get the syndrome vector
 - o 8 times, for each row in H transpose, use bitwise manipulation to get the the value for the syndrome vector, and if it corresponds to a value in H transpose, the error is in the location at the row value in H transpose.

Section 3: Generator.c and Decoder.c

- After initializing the G and H matrices, Receive input via fgetc() and apply ham_encode and ham_decode respectively.
- For decoder.c, create variables for printing Ham_Err,
 Ham err ok, and Ham ok to print to output.