MEMORANDUM

TO: ECE130 Spring 2003-2004

FROM: Laflen

SUBJECT: Request for an Enigma ASCII Encryption/Decryption Device

DATE: 6 May 2004

This memo describes an assignment for you to design, construct, and test an Enigma ASCII Encryption/Decryption Device using an appropriate simulator. The Enigma Device encrypts an ASCII character to another ASCII character according to specifications below. The encryption process is also a decryption process, so any encrypted output will become decrypted when run back through the device.

Desired features of the device, manpower allocation, schedule, report requirement, and evaluation criteria are outlined below.

A Design and Verification of the Enigma Device

The Enigma Device is a programmable ASCII encryption/decryption circuit.

A.1 Interface

A.1.1 Inputs

(1) **CLK:** this is the master clock input to the circuit

   a. Positive (rising) edge triggered

(2) **RESET:** this is the master, asynchronous reset for the circuit

   a. Active high

(3) **LOAD_IN:** this signal indicates a load to an internal variable

   a. Active high

(4) **NEW_IN:** this signal indicates a new ASCII input

   a. Active high

(5) **DIR:** indicates the encryption direction

   a. 0 → encrypt

   b. 1 → decrypt

(6) **ASCII_IN[7..0]:** this is an 8-bit data bus containing the ASCII character to encrypt. ASCII_IN[7] is the MSB (most significant bit) and ASCII_IN[0] is the LSB (least significant bit).

   a. **See specification, below**

A.1.2 Outputs

(1) **LOAD_OUT:** indicates when a load occurred

   a. Active high

(2) **NEW_OUT:** indicates when a new ASCII output has occurred

b. Active high

(3) **ASCII_OUT[7..0]:** the encrypted/decrypted data bundle

c. **See specification, below**

A.2 ASCII Character Set

Below is a table listing the 256 ASCII characters:

Hex	Char	Visible	Notes
0	NUL		Null Character
1	SOH		Cursor Home
2	STX		
3	ETX		
4	EOT		
5	ENQ		
6	ACK		Cursor Forward
7	BEL		Bell
8	BS		Erase Last Character
9	HT		
A	LF		Cursor Down/Line Feed
B	VT		Vertical Address
C	FF		Form Feed/Screen Erase
D	CR		Carriage Return
E	SO		
F	SI		
10	DLE		Horizontal Address
11	DC1		X-ON
12	DC2		Enable Auxiliary Port
13	DC3		X-OFF
14	DC4		Disable Auxiliary Port
15	NAK		Cursor Back
16	SYN		
17	ETB		
18	CAN		
19	EM		
1A	SUB		Cursor Up
1B	ESC		
1C	FS		
1D	GS		
1E	RS		
1F	US		
20	SPC		
21	!	!	
22	"	"	
23	#	#	
24	$	$	
25	%	%	
26	&	&	
27	'	'	
28	(	(	
29	)	)	
2A	*	*	
2B	+	+	
2C	,	,	
2D	-	-	
2E	.	.	
2F	/	/	
30	0	0	
31	1	1	
32	2	2	
33	3	3	
34	4	4	
35	5	5	
36	6	6	
37	7	7	
38	8	8	
39	9	9	
3A	:	:	
3B	;	;	
3C	<	<	
3D	=	=	
3E	>	>	
3F	?	?	
40	@	@	
41	A	A	
42	B	B	
43	C	C	
44	D	D	
45	E	E	
46	F	F	
47	G	G	
48	H	H	
49	I	I	
4A	J	J	
4B	K	K	
4C	L	L	
4D	M	M	
4E	N	N	
4F	O	O	
50	P	P	
51	Q	Q	
52	R	R	
53	S	S	
54	T	T	
55	U	U	
56	V	V	
57	W	W	
58	X	X	
59	Y	Y	
5A	Z	Z	
5B	[	[	
5C	\	\	
5D	]	]	

5E	^	^	
5F			
60	`	`	
61	a	a	
62	b	b	
63	c	c	
64	d	d	
65	e	e	
66	f	f	
67	g	g	
68	h	h	
69	i	i	
6A	j	j	
6B	k	k	
6C	l	l	
6D	m	m	
6E	n	n	
6F	o	o	
70	p	p	
71	q	q	
72	r	r	
73	s	s	
74	t	t	
75	u	u	
76	v	v	
77	w	w	
78	x	x	
79	y	y	
7A	z	z	
7B	{	{	
7C	|	|	
7D	}	}	
7E	~	~	
7F	Del	•	
80	M-^@	Ç	
81	M-^A	ü	
82	M-^B	é	
83	M-^C	â	
84	M-^D	ä	
85	M-^E	à	
86	M-^F	å	
87	M-^G	ç	
88	M-^H	ê	
89	M-^I	ë	
8A	M-^J	è	
8B	M-^K	ï	
8C	M-^L	î	
8D	M-^M	ì	
8E	M-^N	Ä	
8F	M-^O	Å	
90	M-^P	É	
91	M-^Q	æ	
92	M-^R	Æ	
93	M-^S	ô	
94	M-^T	ö	
95	M-^U	ò	
96	M-^V	û	
97	M-^W	ù	
98	M-^X	ÿ	
99	M-^Y	Ö	
9A	M-^Z	Ü	
9B	M-^[	¢	
9C	M-^\	£	
9D	M-^]	¥	
9E	M-^^	•	
9F	M-^_	ƒ	
A0	M-		
A1	M-!	í	

A2	M-"	ó	
A3	M-#	ú	
A4	M-$	ñ	
A5	M-%	Ñ	
A6	M-&	ª	
A7	M-'	º	
A8	M-(	¿	
A9	M-)	•	
AA	M-*	¬	
AB	M-+	½	
AC	M-,	¼	
AD	M--	¡	
AE	M-.	«	
AF	M-/	»	
B0	M-0	•	
B1	M-1	•	
B2	M-2	•	
B3	M-3	•	
B4	M-4	•	
B5	M-5	•	
B6	M-6	•	
B7	M-7	•	
B8	M-8	•	
B9	M-9	•	
BA	M-:	•	
BB	M-;	•	
BC	M-<	•	
BD	M-=	•	
BE	M->	•	
BF	M-?	•	
C0	M-@	•	
C1	M-A	•	
C2	M-B	•	
C3	M-C	•	
C4	M-D	•	
C5	M-E	•	
C6	M-F	•	
C7	M-G	•	
C8	M-H	•	
C9	M-I	•	
CA	M-J	•	
CB	M-K	•	
CC	M-L	•	
CD	M-M	•	
CE	M-N	•	
CF	M-O	•	
D0	M-P	•	
D1	M-Q	•	
D2	M-R	•	
D3	M-S	•	
D4	M-T	•	
D5	M-U	•	
D6	M-V	•	
D7	M-W	•	
D8	M-X	•	
D9	M-Y	•	
DA	M-Z	•	
DB	M-[	•	
DC	M-\	•	
DD	M-]	•	
DE	M-^	•	
DF	M-	•	
E0	M-`	•	
E1	M-a	ß	
E2	M-b	•	
E3	M-c	•	
E4	M-d	•	
E5	M-e	•	

E6	M-f	µ	
E7	M-g	•	
E8	M-h	•	
E9	M-i	•	
EA	M-j	•	
EB	M-k	•	
EC	M-l	•	
ED	M-m	•	
EE	M-n	•	
EF	M-o	•	
F0	M-p	•	
F1	M-q	±	
F2	M-r	•	
F3	M-s	•	
F4	M-t	•	
F5	M-u	•	
F6	M-v	÷	

F7	M-w	•	
F8	M-x	°	
F9	M-y	•	
FA	M-z	·	
FB	SB	•	Start Buffer
FC	SVM	•	SubValue Mark
FD	VM	²	Value Mark
FE	AM	•	Attribute Mark
FF	SM		Segment Mark

A.3 Encryption Specification

> "…Hitler began rearming Germany, and the cryptologic experts of the Wehrmacht, deciding that the Enigma offered satisfactory guarantees of security, began supplying their expanding forces with it. …During World War II, the portable glowlamp Enigma, battery-powered, and, in its wooden box, about the size and weight of a standard typewriter, served as the top German Army, Navy, Air Force system. Signal officers regarded it as very dependable and believed it to be secure. Its only disadvantage was that it did not print, and speedy operation required three men—one to read the incoming text and press the keys, one to call out the letters in a loud voice as they lit up, one to write down the text."
>
> *from David Kahn's The Codebreakers: The Story of Secret Writing.*

The ASCII Enigma Device will be an electronic improvement on the historic Enigma rotor device used during World War II. The device will have two encryption modes: alpha-numeric and non-alpha-numeric. At the beginning stage of the encryption process, the device must decide if the input ASCII code is an alpha-numeric code (*i.e.*, hexadecimal 30 through 7A). If the ASCII code is alpha-numeric, it will be processed in the alpha-numeric mode; otherwise it will be processed in the non-alpha-numeric mode.

A.3.1 Alpha-Numeric Encryption

The device must maintain four 6-bit memory blocks for the encryption algorithm. Two of these cells, labeled **ALPHA** and **OMEGA** will be used to translate the ASCII code as follows:

Let C be the hexadecimal value of an 8-bit alpha-numeric ASCII code

The encrypted value is:

$$E = \textbf{mod}(C - 0x30 + \textbf{ALPHA} - \textbf{OMEGA}, 0x4B) + 0x30$$

The **mod**(***a***, ***b***) operation computes the remainder of ***a*** divided by ***b***. If ***a*** is a negative number, add 0x4B first to make it positive, then compute the remainder. The above encryption operation finds a new alpha-numeric value for C – that is, the output will remain alpha-numeric.

The two other memory blocks are **ALPHAinc** and **OMEGAdec**. After each character encryption, **ALPHA** and **OMEGA** are updated as follows:

ALPHA = **ALPHA** + **ALPHAinc**
OMEGA = **OMEGA** – **OMEGAdec**

To reverse this process, the following is computed:

ALPHA = **ALPHA** – **ALPHAinc**
OMEGA = **OMEGA** + **OMEGAdec**

C = **mod**(E – 0x30 – **ALPHA** + **BETA**, 0x4B) + 0x30

The resulting value (either from encryption or decryption) is output on ASCII_OUT.

A.3.2 Non-Alpha-Numeric Encryption

The non-alpha-numeric encryption is less rigorous. Let T be the inverse of non-alpha-numeric input ASCII_IN (*e.g.*, T6 = ASCII_IN6'). Then:

C	=	T,	if T is non-alpha-numeric
		ASCII_IN,	if T is alpha-numeric

The exact same process decrypts C.

The resulting value is output on ASCII_OUT.

A.4 Load Specification

The **LOAD_IN** signal specifies additional functionality for updating the four 6-bit memory cells for alpha-numeric encryption. If **LOAD_IN** is high, the 8-bit ASCII_IN code is used to update the memory cells and is not processed as an encryption/decryption code.

The memory cell update depends on the two most significant bits of ACII_IN as follows:

ASCII_IN7	ASCII_IN6	Memory Cell
0	0	**ALPHA**
0	1	**ALPHAinc**
1	0	**OMEGA**
1	1	**OMEGAdec**

Depending on the memory cell selected by ASCII_IN[7..6], that memory cell should update (*i.e.*, load) the 6-bit value from ASCII_IN[5..0].

Further, when a memory cell update occurs, the **OLD** value from the memory cell should be output on ASCII_OUT, along with the correct memory cell identification code.

For example,

> LOAD_IN is high and ASCII_IN is 0x87 (binary 1000 0111)
> LOAD_OUT should also be high
> Memory cell **OMEGA** should load the value 000111.
> If **OMEGA** previously had the value 101010,
> then ASCII_OUT should be 0xAA (binary 1010 1010).

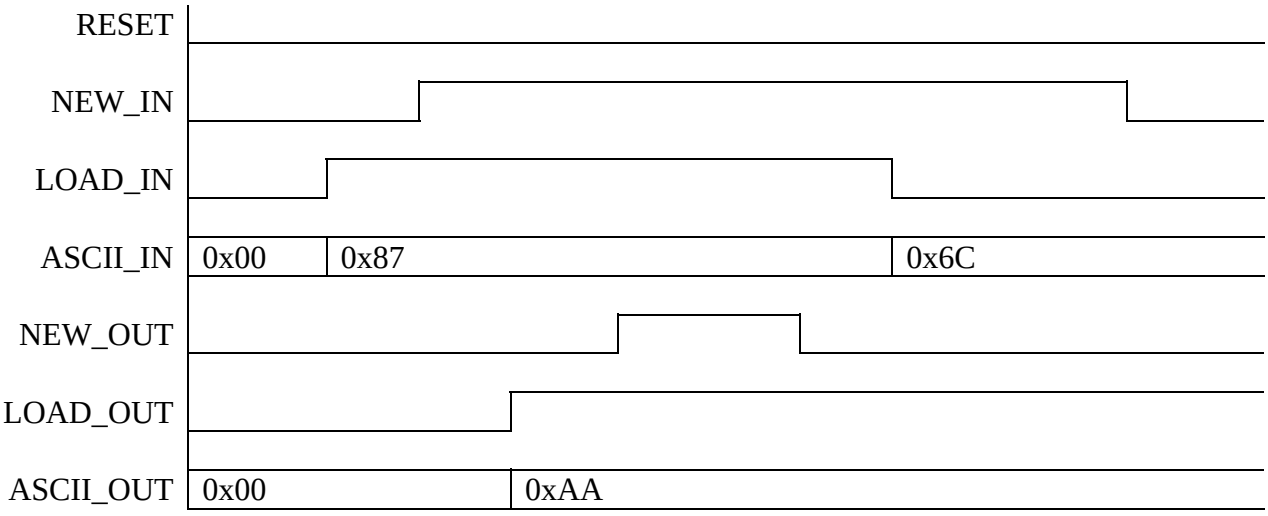
A.5 NEW Signal Specification

The Enigma circuit should only process the ASCII_IN and LOAD_IN signals whenever the NEW_IN signal is asserted. This means that the circuit must wait for NEW_IN to go high, process one ASCII_IN packet, and then wait for NEW_IN to go low before starting the process over.

NOTE: only one ASCII_IN packet should be processed for each rising edge of NEW_IN.

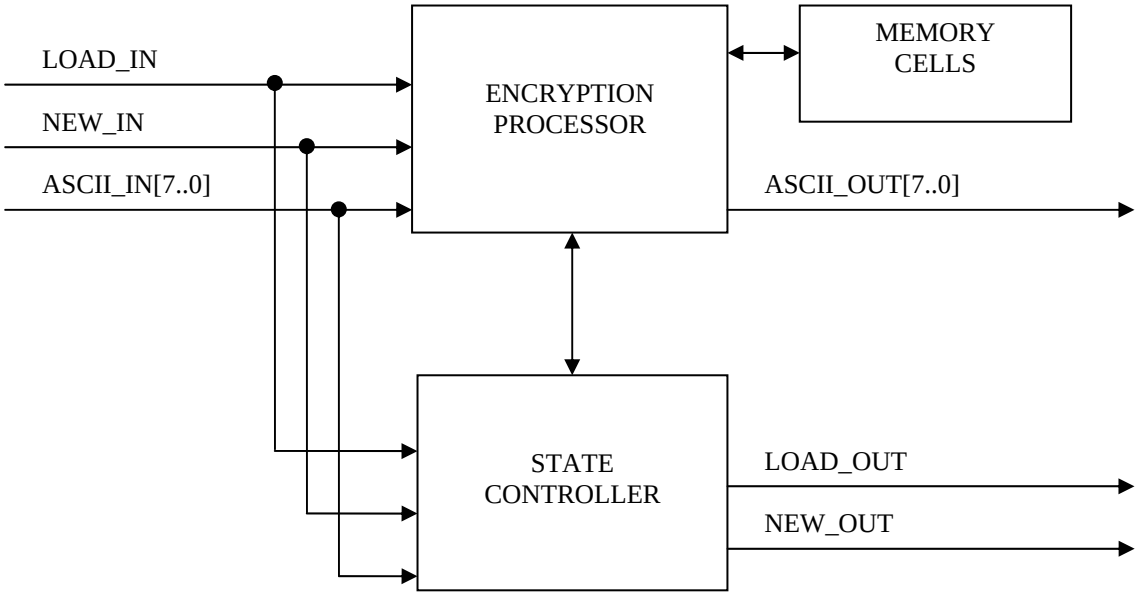
When a new ASCII_IN packet is processed, NEW_OUT should be asserted briefly *AFTER* ASCII_OUT is updated with the correct output signal.

Drawing on the above example in A.4:



A.6 Top-Level Block Diagram

The following is a block diagram to show my conceptual design of the controller. You are free to come up with your own design that should have the same functionality.



B Manpower Allocation

A team of four engineers is assigned to work on the project. Each engineer is required to spend 10 hours per week on average. Therefore, a total of about 80 hours of engineer's time is allocated to the project.

It is up to the team leader to decide how to distribute tasks to each member of the team and/or how to divide the team into groups. The team leader will be responsible for the overall project: scheduling, manpower management, reports, and major decisions, etc.

C Schedule and reports

Four reports are required from each team: three progress reports and one final report. Your progress reports should have detailed descriptions of the required topics as well as a **cover page** that includes the following as shown in a sample report distributed with this memo:

1. Section # and Team #
2. Work completed
3. Current work
4. Future work
5. Red flags (anything that might cause major problems)

Your reports should include major components and diagrams and schematics. They will be evaluated mainly on clarity and completeness. Diagrams can be hand-drawn as long as they are clear and accurate.

A soft copy of the proposed final report template is on the website.

Here is a list of expected results. All of the reports are due by the end of the day and will carry a 20% late penalty for each additional day. Either a hard copy or a soft copy (e-mail) of a report will be accepted.

(1) Selection of the team leader and group leaders (if a team is divided into groups) are due by Monday, May 10. Names of the leaders should be emailed to me.

(2) An initial progress report on project definition, plan and schedule that is due on Tuesday, May 11. This report includes Sections 2 to 4 of the proposed final project report attached to this memo. This report MUST be hand written. (Leave Section 1 Executive Summary until the completion of the project.)

(3) A design progress report on state diagrams and circuit designs for the modules is due on Friday, May 14. This progress report should include Sections 2 to 6 of the draft report. This report MUST be typed. Include your circuit snapshots in this report. Your circuit designs do not have to be verified completely.

(4) An accumulative progress report on everything up to module circuit verification procedures and results on Tuesday, May 18. This report should include Section 2 to 7 of the draft progress report.

The following snapshots are required for each circuit verification: (a) Test circuit with the circuit schematic and test I/O devices that show how the circuit works; (b) the subcircuit with I/O ports (single and bus ports). The subcircuit should not be part of the test circuit but a separate file; (c) Test circuit for the device symbol that shows how the device works. (c) Annotated test waveforms.

(5) The final project report is due on Thursday, May 20. This report should contain all of the sections.

(6) An oral presentation for this project is scheduled during the class periods on Thursday and Friday, May 20-21. A soft copy of your simulation file(s) is due at the same time.

Here are the tasks and deadlines in a bar chart.

Date/Task and deadline	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
Leader selection															
Project definition & plan															
State diagram and circuit design															
Module verification report															
Final project report															
Oral presentation															

D Mandatory Attendance

Every student is required to work on the project in the classroom during lecture hours. An attendance check-off sheet may be circulated in class to record student presence. An individual student's project grade will be that of the group's grade multiplied by his or her attendance percentage.

E Grading Policy

The detailed grading policy is attached to this memo.