The assembler for mic1 (masm) accepts strings of the form:

```
str:   "hi mom"

str1:  "hi moma"
```

and will generate output into the assembly file as:

```
str:    0110100101101000   ←  left byte: i  right byte: h
        0110110100100000   ←  left byte: m  right byte: space
        0110110101101111   ←  left byte: m  right byte: o
        0000000000000000   ←  left byte: \0 right byte: \0

str1:   0110100101101000   ←  left byte: i  right byte: h
        0110110100100000   ←  left byte: m  right byte: space
        0110110101101111   ←  left byte: m  right byte: o
        0000000001100001   ←  left byte: \0 right byte: a
```

This means that bytes are packed two per word with the last word either all full or half full of zeros as shown in the two different cases above. Since we discuss strings in this assignment it is important to understand how they are deployed in memory.

Hex	Dec	Char		Hex	Dec	Char	Hex	Dec	Char	Hex	Dec	Char
0x00	0	NULL	null	0x20	32	Space	0x40	64	@	0x60	96	`
0x01	1	SOH	Start of heading	0x21	33	!	0x41	65	A	0x61	97	a
0x02	2	STX	Start of text	0x22	34	"	0x42	66	B	0x62	98	b
0x03	3	ETX	End of text	0x23	35	#	0x43	67	C	0x63	99	c
0x04	4	EOT	End of transmission	0x24	36	$	0x44	68	D	0x64	100	d
0x05	5	ENQ	Enquiry	0x25	37	%	0x45	69	E	0x65	101	e
0x06	6	ACK	Acknowledge	0x26	38	&	0x46	70	F	0x66	102	f
0x07	7	BELL	Bell	0x27	39	'	0x47	71	G	0x67	103	g
0x08	8	BS	Backspace	0x28	40	(	0x48	72	H	0x68	104	h
0x09	9	TAB	Horizontal tab	0x29	41	)	0x49	73	I	0x69	105	i
0x0A	10	LF	New line	0x2A	42	*	0x4A	74	J	0x6A	106	j
0x0B	11	VT	Vertical tab	0x2B	43	+	0x4B	75	K	0x6B	107	k
0x0C	12	FF	Form Feed	0x2C	44	,	0x4C	76	L	0x6C	108	l
0x0D	13	CR	Carriage return	0x2D	45	-	0x4D	77	M	0x6D	109	m
0x0E	14	SO	Shift out	0x2E	46	.	0x4E	78	N	0x6E	110	n
0x0F	15	SI	Shift in	0x2F	47	/	0x4F	79	O	0x6F	111	o
0x10	16	DLE	Data link escape	0x30	48	0	0x50	80	P	0x70	112	p
0x11	17	DC1	Device control 1	0x31	49	1	0x51	81	Q	0x71	113	q
0x12	18	DC2	Device control 2	0x32	50	2	0x52	82	R	0x72	114	r
0x13	19	DC3	Device control 3	0x33	51	3	0x53	83	S	0x73	115	s
0x14	20	DC4	Device control 4	0x34	52	4	0x54	84	T	0x74	116	t
0x15	21	NAK	Negative ack	0x35	53	5	0x55	85	U	0x75	117	u
0x16	22	SYN	Synchronous idle	0x36	54	6	0x56	86	V	0x76	118	v
0x17	23	ETB	End transmission block	0x37	55	7	0x57	87	W	0x77	119	w
0x18	24	CAN	Cancel	0x38	56	8	0x58	88	X	0x78	120	x
0x19	25	EM	End of medium	0x39	57	9	0x59	89	Y	0x79	121	y
0x1A	26	SUB	Substitute	0x3A	58	:	0x5A	90	Z	0x7A	122	z
0x1B	27	FSC	Escape	0x3B	59	;	0x5B	91	[	0x7B	123	{
0x1C	28	FS	File separator	0x3C	60	<	0x5C	92	\	0x7C	124	|
0x1D	29	GS	Group separator	0x3D	61	=	0x5D	93	]	0x7D	125	}
0x1E	30	RS	Record separator	0x3E	62	>	0x5E	94	^	0x7E	126	~
0x1F	31	US	Unit separator	0x3F	63	?	0x5F	95	_	0x7F	127	DEL

ASCII Character Code Chart

MJ Karas

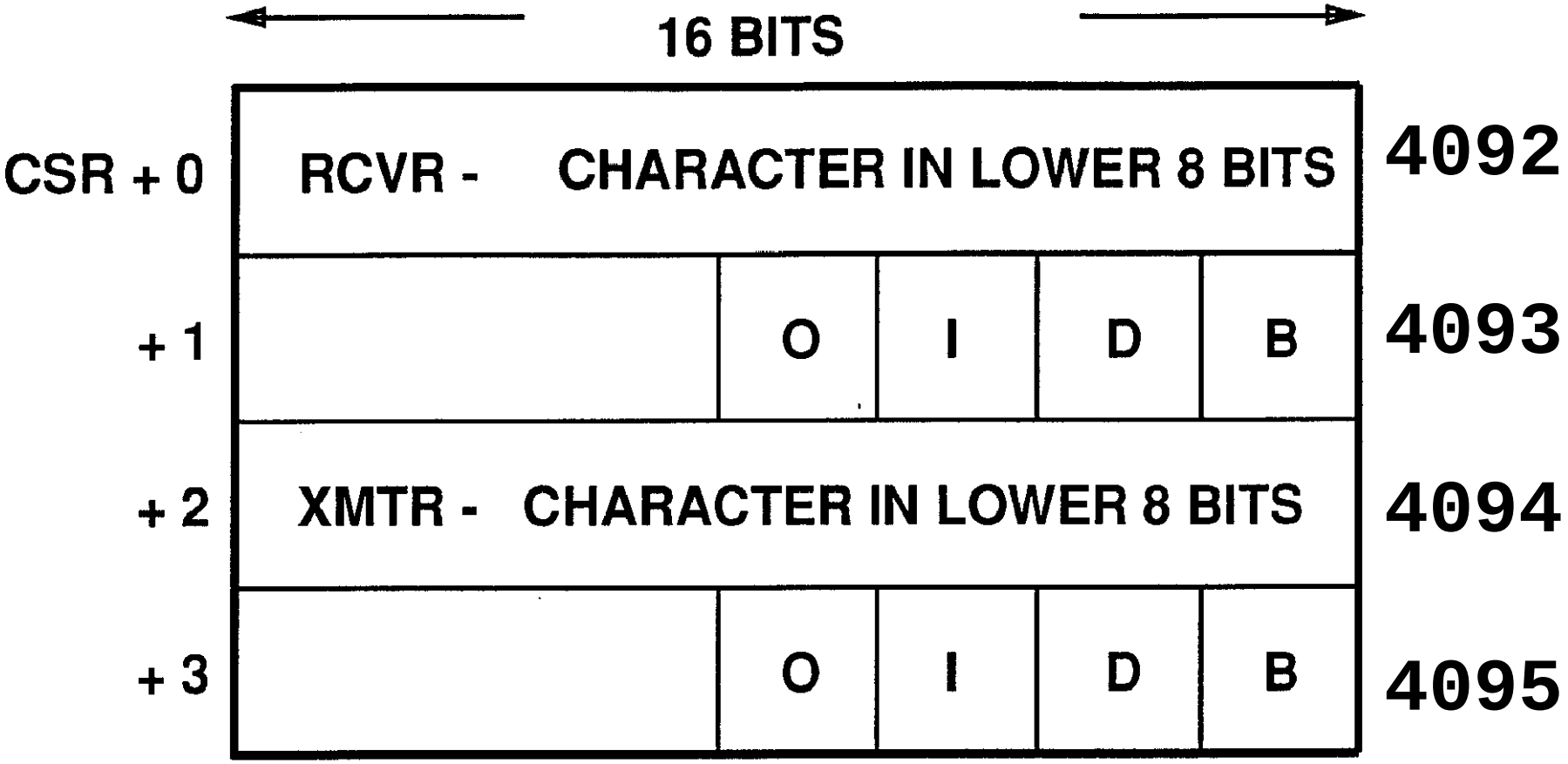
Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char
0	00	000	NUL	32	20	040	SP	64	40	100	@	96	60	140	`
1	01	001	SOH	33	21	041	!	65	41	101	A	97	61	141	a
2	02	002	STX	34	22	042	"	66	42	102	B	98	62	142	b
3	03	003	ETX	35	23	043	#	67	43	103	C	99	63	143	c
4	04	004	EOT	36	24	044	$	68	44	104	D	100	64	144	d
5	05	005	ENQ	37	25	045	%	69	45	105	E	101	65	145	e
6	06	006	ACK	38	26	046	&	70	46	106	F	102	66	146	f
7	07	007	BEL	39	27	047	'	71	47	107	G	103	67	147	g
8	08	010	BS	40	28	050	(	72	48	110	H	104	68	150	h
9	09	011	TAB	41	29	051	)	73	49	111	I	105	69	151	i
10	0A	012	LF	42	2A	052	*	74	4A	112	J	106	6A	152	j
11	0B	013	VT	43	2B	053	+	75	4B	113	K	107	6B	153	k
12	0C	014	FF	44	2C	054	,	76	4C	114	L	108	6C	154	l
13	0D	015	CR	45	2D	055	-	77	4D	115	M	109	6D	155	m
14	0E	016	SO	46	2E	056	.	78	4E	116	N	110	6E	156	n
15	0F	017	SI	47	2F	057	/	79	4F	117	O	111	6F	157	o
16	10	020	DLE	48	30	060	0	80	50	120	P	112	70	160	p
17	11	021	DC1	49	31	061	1	81	51	121	Q	113	71	161	q
18	12	022	DC2	50	32	062	2	82	52	122	R	114	72	162	r
19	13	023	DC3	51	33	063	3	83	53	123	S	115	73	163	s
20	14	024	DC4	52	34	064	4	84	54	124	T	116	74	164	t
21	15	025	NAK	53	35	065	5	85	55	125	U	117	75	165	u
22	16	026	SYN	54	36	066	6	86	56	126	V	118	76	166	v
23	17	027	ETB	55	37	067	7	87	57	127	W	119	77	167	w
24	18	030	CAN	56	38	070	8	88	58	130	X	120	78	170	x
25	19	031	EM	57	39	071	9	89	59	131	Y	121	79	171	y
26	1A	032	SUB	58	3A	072	:	90	5A	132	Z	122	7A	172	z
27	1B	033	ESC	59	3B	073	;	91	5B	133	[	123	7B	173	{
28	1C	034	FS	60	3C	074	<	92	5C	134	\	124	7C	174	|
29	1D	035	GS	61	3D	075	=	93	5D	135	]	125	7D	175	}
30	1E	036	RS	62	3E	076	>	94	5E	136	^	126	7E	176	~
31	1F	037	US	63	3F	077	?	95	5F	137	_	127	7F	177	DEL

Byte	Cod.	Char
00000000	0	Null
00000001	1	Start of heading
00000010	2	Start of text
00000011	3	End of text
00000100	4	End of transmit
00000101	5	Enquiry
00000110	6	Acknowledge
00000111	7	Audible bell
00001000	8	Backspace
00001001	9	Horizontal tab
00001010	10	Line feed
00001011	11	Vertical tab
00001100	12	Form Feed
00001101	13	Carriage return
00001110	14	Shift out
00001111	15	Shift in
00010000	16	Data link escape
00010001	17	Device control 1
00010010	18	Device control 2
00010011	19	Device control 3
00010100	20	Device control 4
00010101	21	Neg. acknowledge
00010110	22	Synchronous idle
00010111	23	End trans. block
00011000	24	Cancel
00011001	25	End of medium
00011010	26	Substitution
00011011	27	Escape
00011100	28	File separator
00011101	29	Group separator
00011110	30	Record Separator
00011111	31	Unit separator

Byte	Cod.	Char
00100000	32	Spc
00100001	33	!
00100010	34	"
00100011	35	#
00100100	36	$
00100101	37	%
00100110	38	&
00100111	39	'
00101000	40	(
00101001	41	)
00101010	42	*
00101011	43	+
00101100	44	,
00101101	45	-
00101110	46	.
00101111	47	/
00110000	48	0
00110001	49	1
00110010	50	2
00110011	51	3
00110100	52	4
00110101	53	5
00110110	54	6
00110111	55	7
00111000	56	8
00111001	57	9
00111010	58	:
00111011	59	;
00111100	60	<
00111101	61	=
00111110	62	>
00111111	63	?

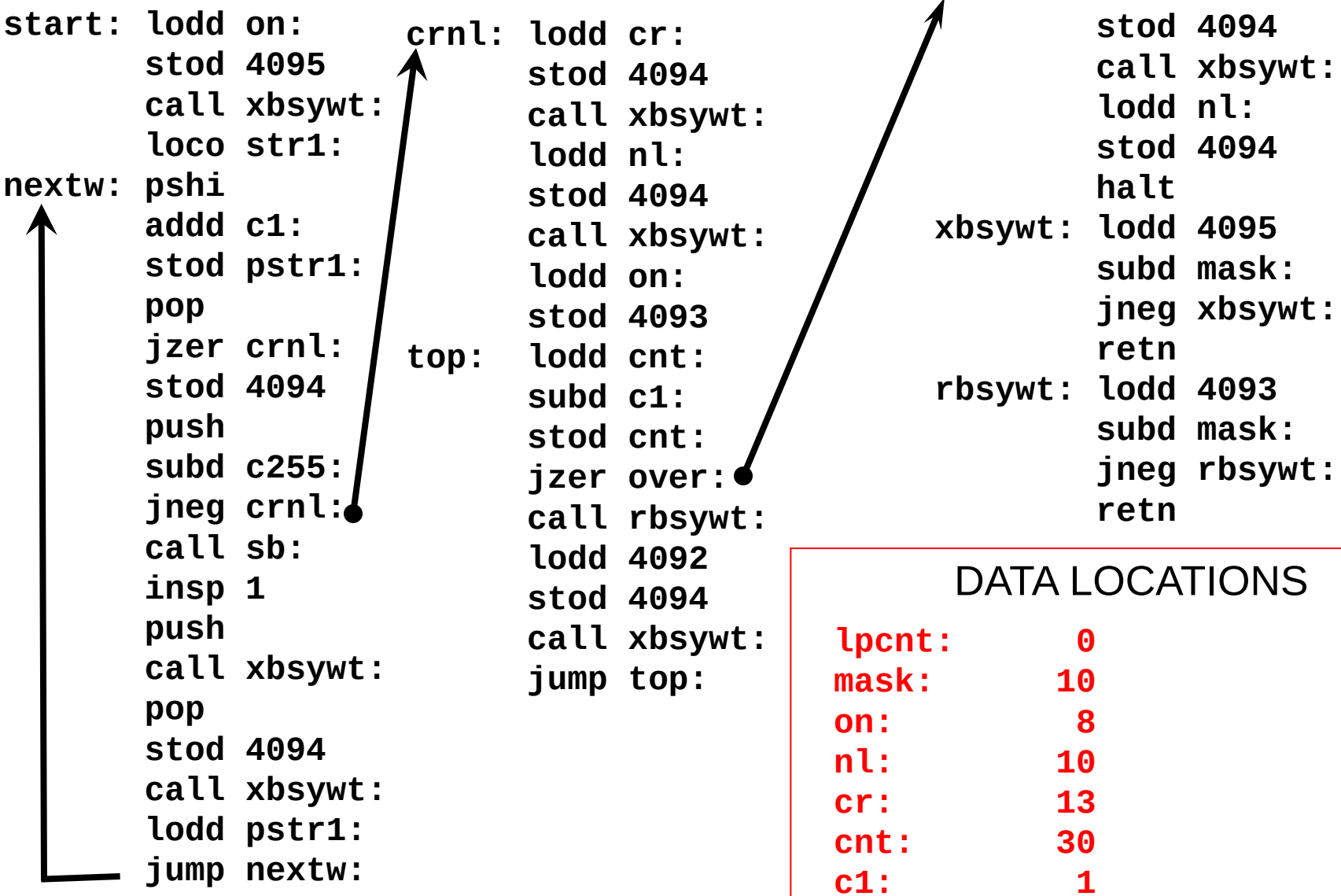
Byte	Cod.	Char
01000000	64	@
01000001	65	A
01000010	66	B
01000011	67	C
01000100	68	D
01000101	69	E
01000110	70	F
01000111	71	G
01001000	72	H
01001001	73	I
01001010	74	J
01001011	75	K
01001100	76	L
01001101	77	M
01001110	78	N
01001111	79	O
01010000	80	P
01010001	81	Q
01010010	82	R
01010011	83	S
01010100	84	T
01010101	85	U
01010110	86	V
01010111	87	W
01011000	88	X
01011001	89	Y
01011010	90	Z
01011011	91	[
01011100	92	\
01011101	93	]
01011110	94	^
01011111	95	_

Byte	Cod.	Char
01100000	96	`
01100001	97	a
01100010	98	b
01100011	99	c
01100100	100	d
01100101	101	e
01100110	102	f
01100111	103	g
01101000	104	h
01101001	105	i
01101010	106	j
01101011	107	k
01101100	108	l
01101101	109	m
01101110	110	n
01101111	111	o
01110000	112	p
01110001	113	q
01110010	114	r
01110011	115	s
01110100	116	t
01110101	117	u
01110110	118	v
01110111	119	w
01111000	120	x
01111001	121	y
01111010	122	z
01111011	123	{
01111100	124	|
01111101	125	}
01111110	126	~
01111111	127	Del


16 BITS
CSR + 0
RCVR - CHARACTER IN LOWER 8 BITS
4092
+ 1
O
I
D
B
4093
+ 2
XMTR - CHARACTER IN LOWER 8 BITS
4094
+ 3
O
I
D
B
4095

The serial line controller for our machine simulation uses the layout shown above. The receiver will accept characters from a terminal device only if it has been turned ON by setting the O bit in the RCVR status register. Turning the receiver on will set B and clear D. When a character has arrived in the RCVR, D will be set and B cleared. If the I bit is set in the RCVR status register then an interrupt will be posted each time the done bit is set. The removal of a character from the RCVR register will clear the interrupt and the done bit and will set busy. If the O bit is cleared the device is effectively turned off, clearing all other bits.

The transmitter must have its O bit set to become active. When O is set the transmitter will immediatlely set done and force an interrupt if I is set. Any characters moved into the XMTR character register at this point will cause D to clear and will clear any outstanding interrupt. Such a charcter will then be drawn on the terminal display and upon completion, the D bit will be set, the B bit cleared and an interrupt will be posted if I is set. Moving another character to the transmitter will clear D and any interrupt as discussed above.



DATA LOCATIONS

```
lpcnt:       0
mask:       10
on:          8
nl:         10
cr:         13
cnt:        30
c1:          1
c255:      255
pstr1:       0
str1:    "THIS IS A TEST STRING1"
```

```
sb:      loco 8
loop1:   jzer finish:
         subd c1:
         stod lpcnt:
         lodl 1
         jneg add1:
         addl 1
         stol 1
         lodd lpcnt:
         jump loop1:
add1:    addl 1
         addd c1:
         stol 1
         lodd lpcnt:
         jump loop1:
finish:  lodl 1
         retn
```

DATA LOCATIONS

```
lpcnt:       0
mask:       10
on:          8
nl:         10
cr:         13
cnt:        30
c1:          1
c255:      255
pstr1:       0
str1:    "THIS IS A TEST STRING1"
```

Ascii Table

DEC	HEX	CH	DEC	HEX	CH	DEC	HEX	CH	DEC	HEX	CH
000	0000	^@	032	0x20	<sp>	064	0x40	@	096	0x60	`
001	0x01	^A	033	0x21	!	065	0x41	A	097	0x61	a
002	0x02	^B	034	0x22	"	066	0x42	B	098	0x62	b
003	0x03	^C	035	0x23	#	067	0x43	C	099	0x63	c
004	0x04	^D	036	0x24	$	068	0x44	D	100	0x64	d
005	0x05	^E	037	0x25	%	069	0x45	E	101	0x65	e
006	0x06	^F	038	0x26	&	070	0x46	F	102	0x66	f
007	0x07	^G	039	0x27	'	071	0x47	G	103	0x67	g
008	0x08	^H	040	0x28	(	072	0x48	H	104	0x68	h
009	0x09	^I	041	0x29	)	073	0x49	I	105	0x69	i
010	0x0a	^J	042	0x2a	*	074	0x4a	J	106	0x6a	j
011	0x0b	^K	043	0x2b	+	075	0x4b	K	107	0x6b	k
012	0x0c	^L	044	0x2c	,	076	0x4c	L	108	0x6c	l
013	0x0d	^M	045	0x2d	-	077	0x4d	M	109	0x6d	m
014	0x0e	^N	046	0x2e	.	078	0x4e	N	110	0x6e	n
015	0x0f	^O	047	0x2f	/	079	0x4f	O	111	0x6f	o
016	0x10	^P	048	0x30	0	080	0x50	P	112	0x70	p
017	0x11	^Q	049	0x31	1	081	0x51	Q	113	0x71	q
018	0x12	^R	050	0x32	2	082	0x52	R	114	0x72	r
019	0x13	^S	051	0x33	3	083	0x53	S	115	0x73	s
020	0x14	^T	052	0x34	4	084	0x54	T	116	0x74	t
021	0x15	^U	053	0x35	5	085	0x55	U	117	0x75	u
022	0x16	^V	054	0x36	6	086	0x56	V	118	0x76	v
023	0x17	^W	055	0x37	7	087	0x57	W	119	0x77	w
024	0x18	^X	056	0x38	8	088	0x58	X	120	0x78	x
025	0x19	^Y	057	0x39	9	089	0x59	Y	121	0x79	y
026	0x1a	^Z	058	0x3a	:	090	0x5a	Z	122	0x7a	z
027	0x1b	^[	059	0x3b	;	091	0x5b	[	123	0x7b	{
028	0x1c	^\	060	0x3c	<	092	0x5c	\	124	0x7c	|
029	0x1d	^]	061	0x3d	=	093	0x5d	]	125	0x7d	}
030	0x1e	^^	062	0x3e	>	094	0x5e	^	126	0x7e	~
031	0x1f	^_	063	0x3f	?	095	0x5f	_	127	0x7f	^?