TI 59 to UDP Interface Software

In order to understand the software, it is first necessary to understand how the Data is formated on the output of the 59 and how the data is translated by the Interface hardware. A brief description is given below.

The Interface Hardware is conected to the data and control lines of the 59 serial interface. It is set up to detect when a load printer command is sent by the 59. This load printer command is sent ON the transfer of each 2 digit pair in response to an OP 05 in the software. The two digit pairs are sent in 2 and 1/3 digit octal (ie 7 bits of data are sent; 1 leading digit plus two 3 bit octal digits) see examples below.

Digit Pair	7BIT Binary Transfer
	0 1 2 3 4 5 6
37	0 | 0 1 1 | 1 1 1
51	0 | 1 0 1 | 0 0 1
15	0 | 0 0 1 | 1 0 1
29*	0 | 0 1 1 | 0 0 1

*The 59 will send Digits ≥ 8 (ie an invalid octal digit) by "carrying out" to the next digit position. We were not shure exactly what would happen for all cases so we always generate valid Octal digits in the software (assuming valid inputs).

The Inteface hardware Ignors Data bit 0 sent by the 59. the other six bits of data are treated as below

Bit 1 - Is used to generate a one shot Write Chip select to tranfer the Data to the UDP if one

Bit 2 - Is goes to the Mode pin.

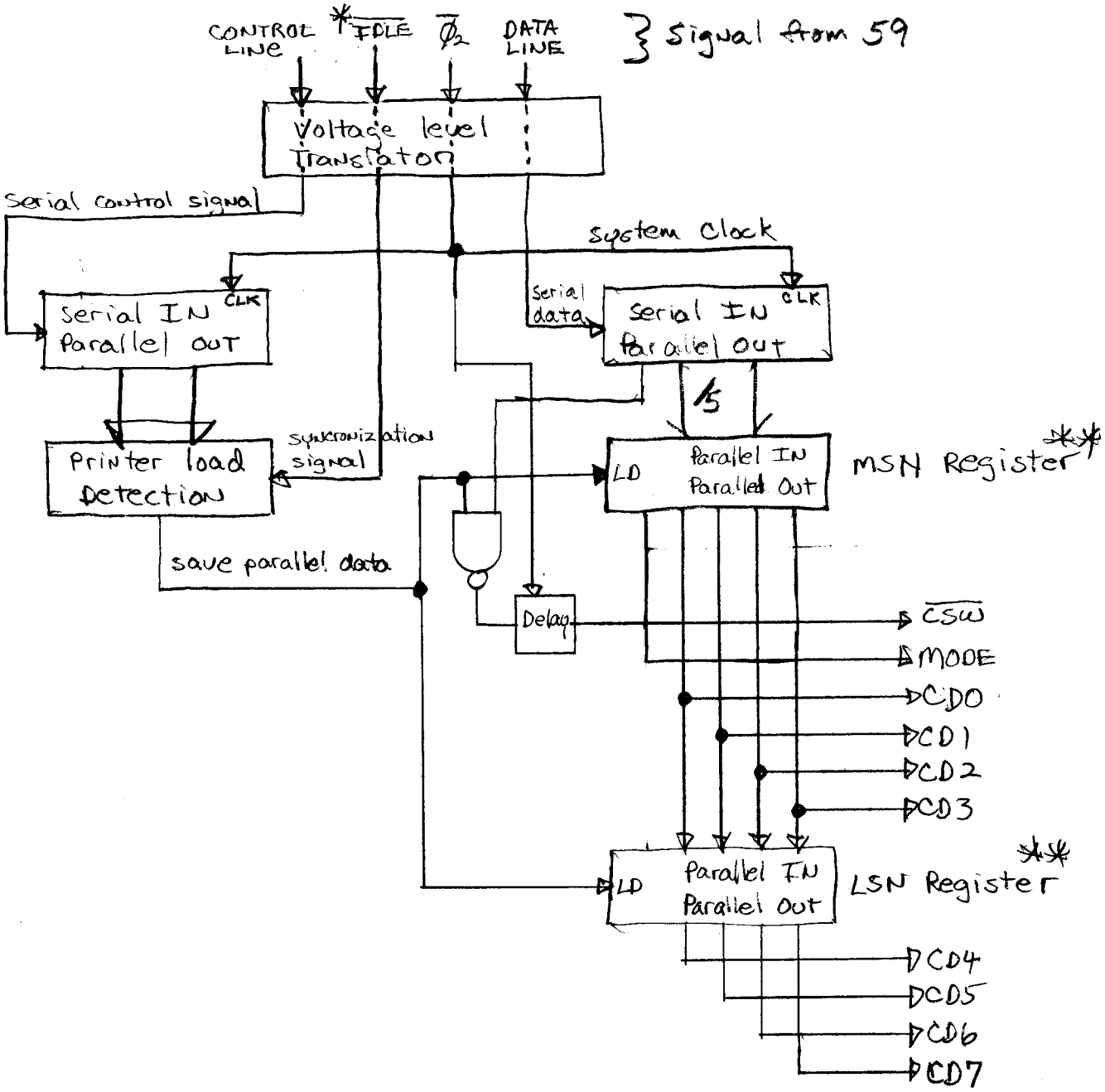
Bits 3 to 6 - are used to send 4 bits of data on each transfer.

Two transfers are required to send a byte of data to the UDP. On the first transfer Bit 1 must be zero, Bit 2 is a "don't care", and Bits 3 to 6 are use to transfer the least significant four bits of data (Least significant Nibble). On the second transfer Bit 1 must be set to ONE, BIT 2 is set to the MODE desired, and Bits 3 to 6 are used to transfer the Most significant 4 bits of data (MSN = Most significant Nibble). How this works to form the control and byte of Data sent to the UDP may be best understood by studying the TI-59 to UDP system on the following page.

The 59 follows a least significant sent first to most significant sent last convention, both at the digit and bit level (for example if the number "4615" is sent a ONE would be sent first corresponding to the least significant bit of the digit "5"). The 59 sends the data loaded by OP Ø4 first followed by OP Ø3, OP Ø2, and OP Ø1 last when a OP Ø5 causes a "PRINT". This ordering is very important to keep in mind since the transfer before the transfer containing a $\overline{CSW}$ forms the data LS Bits and due to the auto-incrementing address register on the UDP that is used to transfer blocks of data.

Examples of how data can be transfered to the UDP are given below

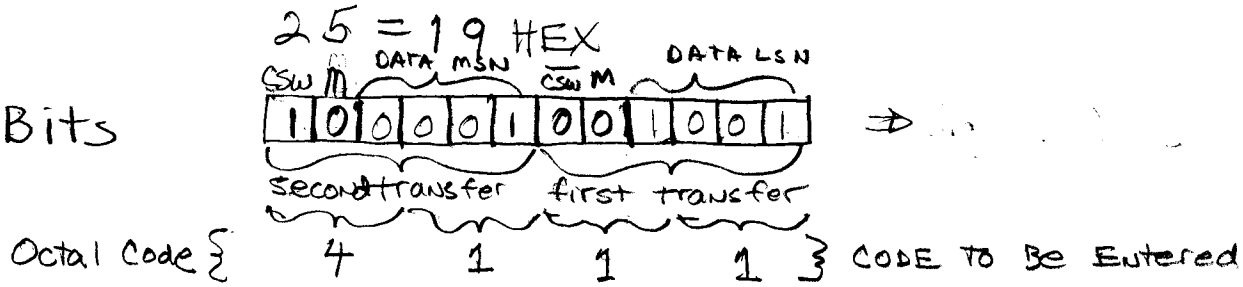
59 to VDP SYSTEM



* Note $\overline{\text{IDLE}}$ is used as a serial data SYNCRONIZATION CONTROL SIGNAL

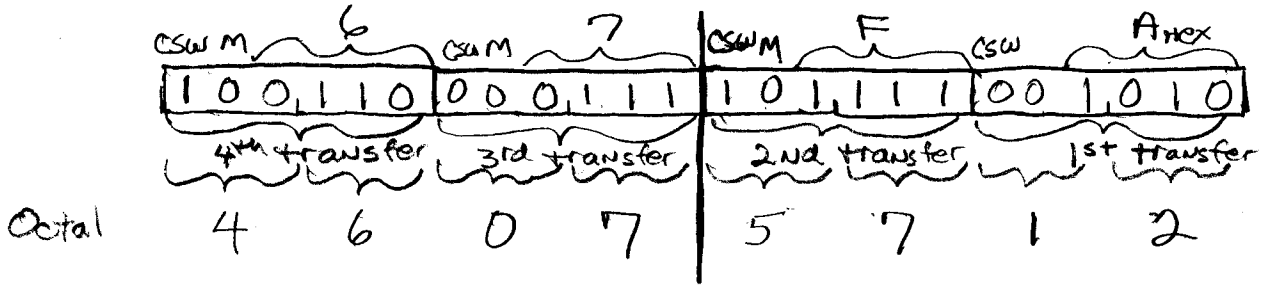
** the MSN - LSN Regs are set up such that after 2 transfers of data they contain the 8 bit data. On the first transfer the MSN contains the least significant bits. On the second transfer the least significant bits are transferred to LSN Reg and the Most significant bits are loaded into MSN Reg.

Example 1 : Write 25 to DRAM at the address Already in the VDP address Register



Example 2 : Write 250 to DRAM at the address Specified IN the VDP address Register followed by writing 1.03 to the next location IN DRAM

250 = FA HEX 103 = 67 Hex



Example 3 : Same Data to DRAM as example 2 but demonstrating how leading or trailing zero's are ignored

Octal Result. 4607005712

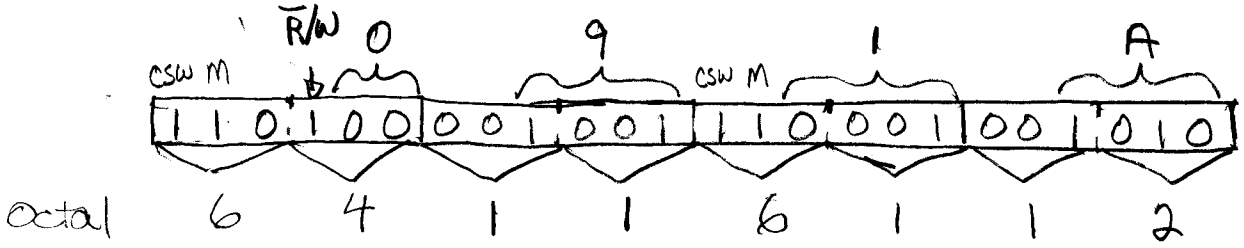
ignored

*Note the two zero's are ignored since the next transfer "07" does NOT have a CSW signal. Thus data WRITES to the VDP and DRAM Really consist of 4 digit groups.

** Note also that the Print registers each hold 10 decimal digits. Thus 2 Bytes of data (requiring 8 digits) can fit in each register

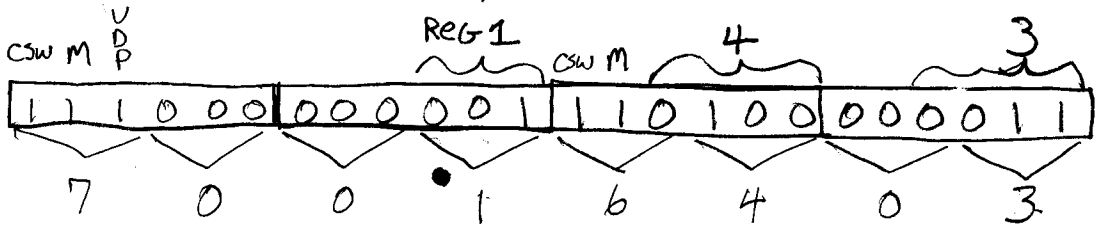
EXAMPLE 4 – Load the VDP address register with 2329 and set the READ/WRITE flag to 1 inside UDP

3330 = 0911A Hex



EXAMPLE 5 – LOAD VDP register 1 with the number 67

67 = 43 Hex



EXAMPLE 6 – Load VDP Register 1 with 67 then write 250, 103, and 25 to locations 3330, 3331 and 3332 Respectively.

ENTER	70016403	OP04	.. Save Number of Example 5 in print Register 4
ENTER	64116112	OP03	.. Save Number of Example 4 in print Register 3
Enter	46075712	OP02	.. Save Number of Example 2
Enter	4111	OP01	.. SAVE Number of Exampl 1
Enter	OP05		.. "Print" to UDP

Converting a number from decimal to the 59-to UDP code is fairly simple. The first step is to convert the number to Hex* and then to Octal and then add on the control bits. Note that the UDP has an 8 bit port and thus the valid number range is 0 to 255, or two hex digits. The basic steps are:

1) The MS Hex "digit"* is simply the integer part of the result of dividing the Number by 16.

2) The Hex "digit" is coverted to OCTAL by adding 2 to the Hex "digit" if and only if the digit is ≥ 8, otherwise the Number is already OK.

3) The control can be added at this point. 40 would set the CSW control. 60 would set the CSW control and cause the data to transfer to the Address Register.

4) Shift the Result to the Left by 2 digit by multiplying by 100. This puts the MS Digit plus control in its proper place

5) The L.S. Hex "digit" is the remainder of the original Divide By 16. The remainder is obtained by multiplying the NON integer part of the Division result by 16.

6) The Hex "digit" is converted to Octal as in step 2.

---

* Note the calculator does not have hex Numbers for A, B, C, D, E, and F so they are represented by the decimal numbers 10, 11, 12, 13, 14 and 15 respectively. By "Hex digit here, it is meant a Number which represents 4 binary bits

7). The LS octal result is added to the result of step 4 to give the final converted result.

Example 7: Convert 250

Step 1: $250 \div 16 = 15.625 \Rightarrow$ MS Hex "Digit" = 15
Step 2: $15 \geq 8$ so $15 + 2 \Rightarrow 17$
Step 3: Add on control to make the Number Write to ORAM $\Rightarrow 17 + 40 \Rightarrow 57$
Step 4: Shift left by 2 $100 \times 57 \Rightarrow 5700$
STEP 5: $.625 \times 16 = 10 \Rightarrow$ LS Hex "Digit" = 10
Step 6: $10 \geq 8$ so $10 + 2 \Rightarrow 12$
Step 7: $5700 + 12 = 5712$ Result.

Note this is the same result as Example 2

Program A' directly impliments the algorithm given above.

INPUT: Decimal Number on display
OUTPUT: Converted Number with Control to write to memory. Result on Display

Program B' is used to generate converted data to write to a UDP register.

INPUT: (Value).(Register Number). The Value must be between 0 and 255 and the Register Number must be ONE digit on the Right of the decimal place
Output: Converted Number with control code for writing to an internal register. The result is "Printed" (or sent) to the UDP and is left on the display.

Example: $67.1 \Rightarrow B' \Rightarrow 70016403$

Program D is used to transfer data that has already been "translated" to the UDP. It transfers the contents of Registers 28 to 59 to the "printer". The Typical Application is that information is translated and STORED IN Registers 30 to 59 (memory Partition #3). The translated data can be Data for DRAM, Addresses, or Internal Register WRITES IN ANY order prouiding it has the correct control signals in the data. (Registers 28 and 29 are transfered but usually will contain Ø. They are only transfered to give an EVEN multiple of 4 transfers). The Data IN Registers 30 to 59 can then be stored ON Mag. cards. The mag. cards containing the translated data are transferred back to Reg 30 to 59 at some later time to write to the UDP

INPUT: Mag Card with pre translated data. Mag card reads into Partition 3.

Output: ON pressing D the contents of Register 28 to 59 are transferred to UDP.

Program D' is used to generate a translated address and load it into the UDP Address register. Note 2 Bytes are transferred to form a 14 bit Address

INPUT: Address between 0 and 16K ON the display

Output: ON pressing D' the address is translated, displayed and shipped to UDP

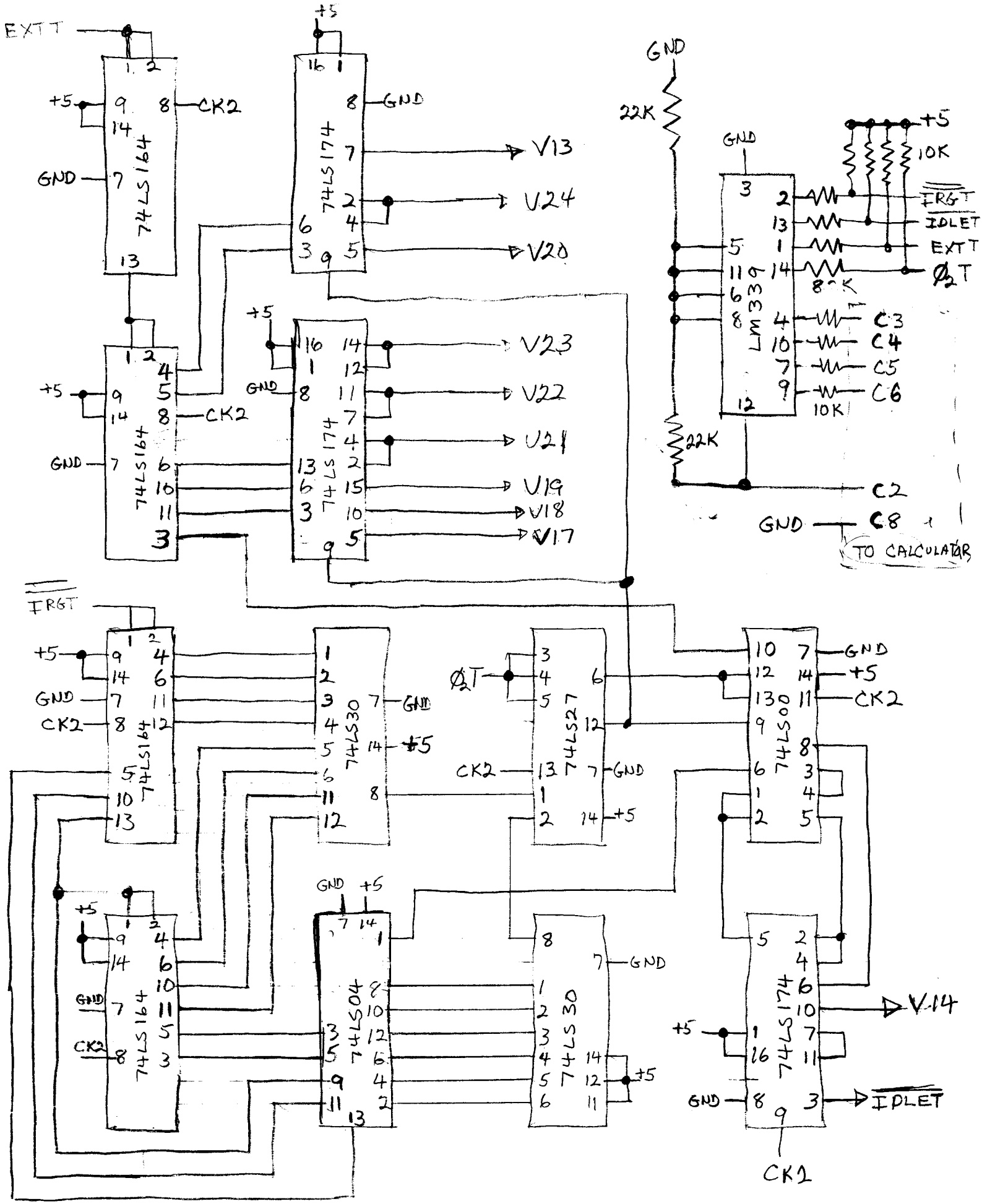
Program E' is used to load the registers 30-59. Registers IN order, It is a simple program used to store translated data. Later this translated data can be used by Program D.

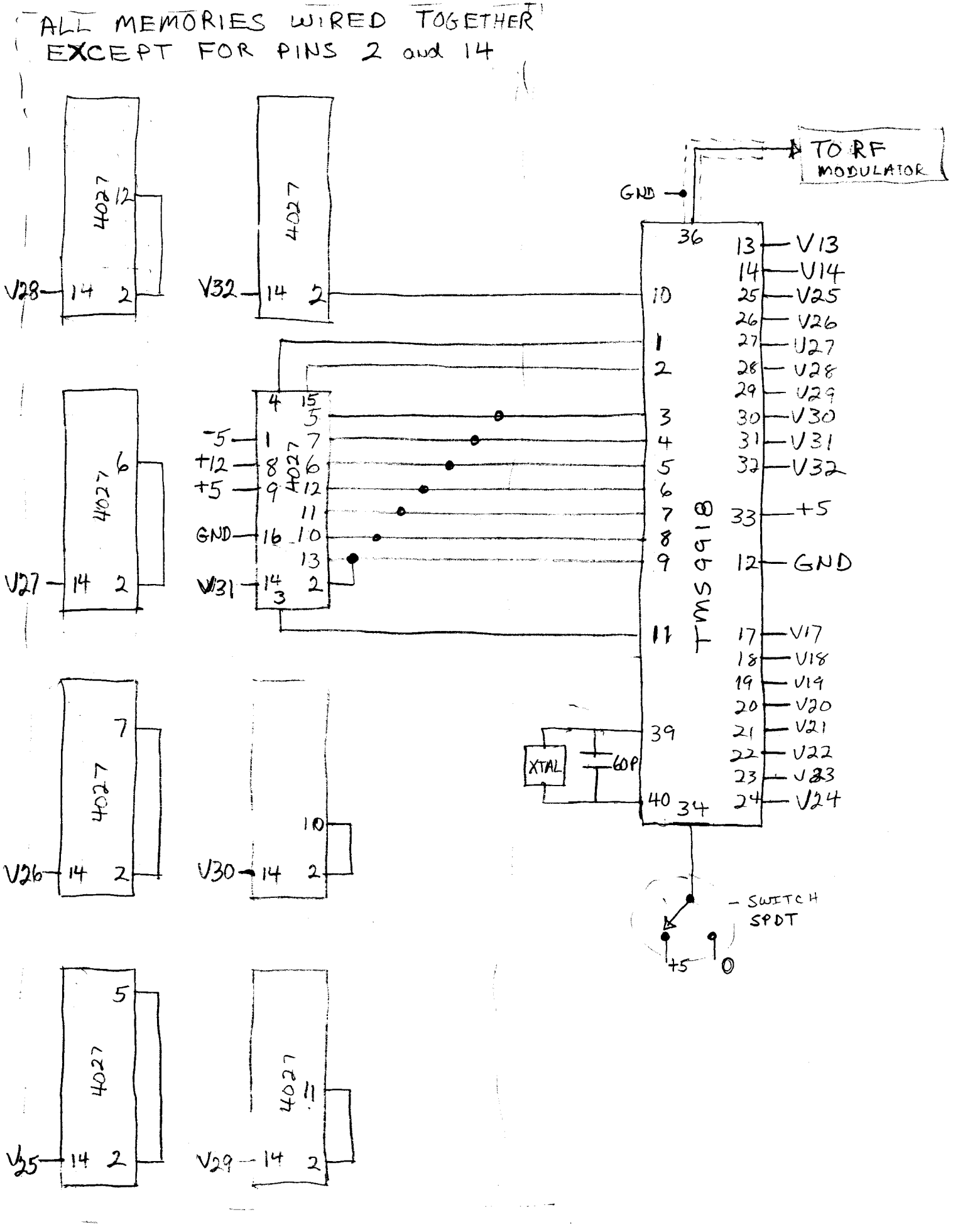
Program C' is use to clear Blocks of Memory in the DRAM.

Inputs: Number of Groups of 16 memory locations to be cleared on Display

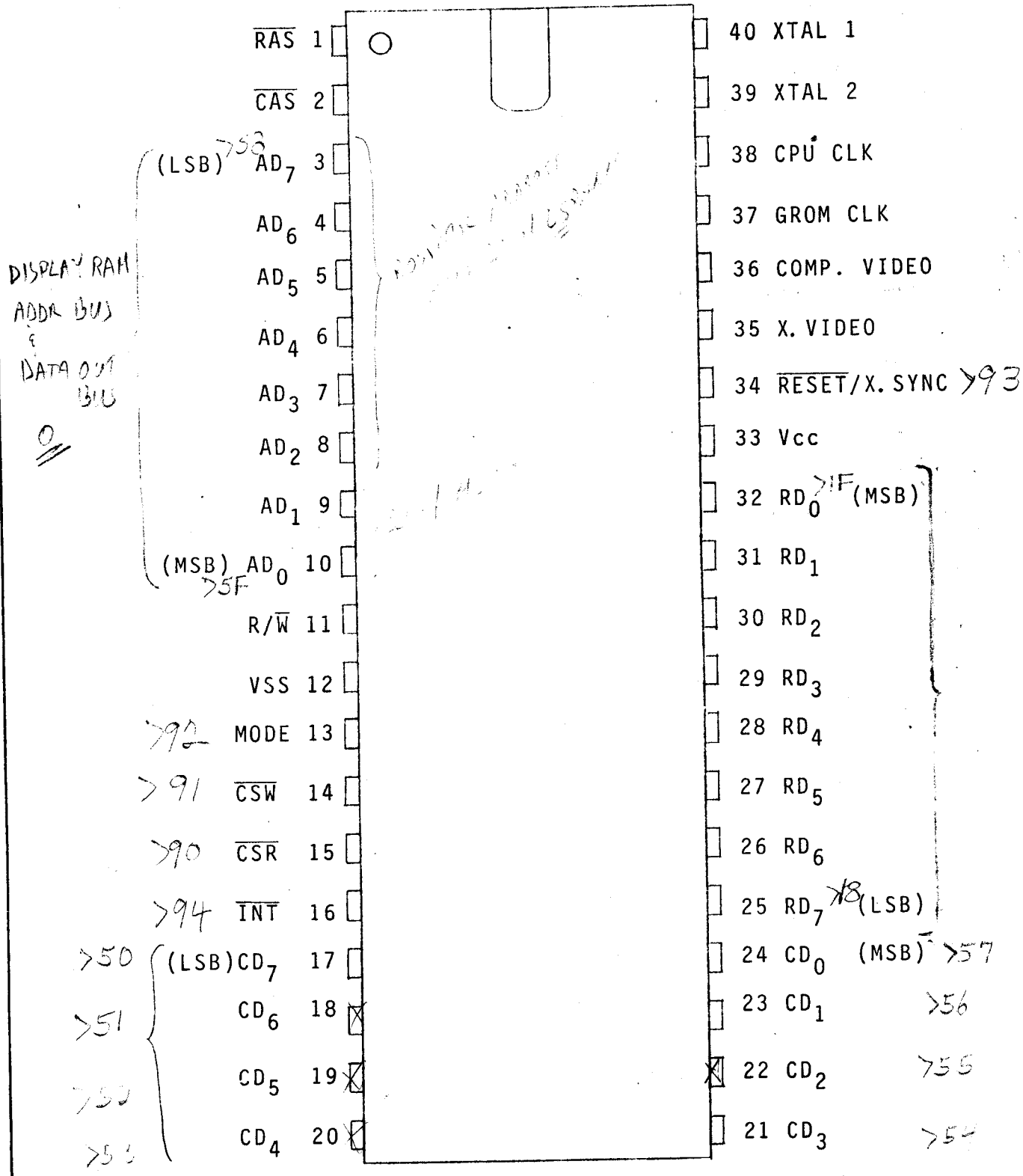
Outputs: Clear Memory locations starting at current content of Address Counter.

Note this program is usually preceeded by Program D' that sets the starting address. The code 4040404000 is used to clear four Bytes of memory; It is a condensed form that takes advantage of how the Harware works. Note that the first 4000 clears the MSN and LSN Register in the harware. Since the next 40 codes keep entering zero's in the data field and cause a cscg a series of zero's are writen to memory.


EXTT
+5
CK2
GND
74LS164
74LS174
V13
V24
V20
V23
V22
V21
V19
V18
V17
22K
LM339
10K
IRGT
IDLET
EXTT
Ø2T
C3
C4
C5
C6
C2
C8
TO CALCULATOR
74LS30
74LS27
74LS00
74LS04
V14
CK1


ALL MEMORIES WIRED TOGETHER
EXCEPT FOR PINS 2 and 14
TO RF MODULATOR
GND
TMS9918
4027
V28
V32
V27
V31
V26
V30
V25
V29
-5
+12
+5
GND
XTAL
60P
SWITCH SPDT
+5
0

TMS 9918 - VDP
PIN ASSIGNMENT



A SIZE	TMS 9918 - VDP	SHEET

TI-11338C

```
70016407
70026000
70036016
70046001
70056006
70066001
70076015
```
INIT. BASE REG

20

80 — 100 GROUPS of 16

70S = GROUPS of 4

6

6

```
64037000
40174117
41174117
41174117
41174117
```
WRITE OUT Color TABLE ADDRES

50

```
64036000
41004008
40064001
46004008
40064004
45004008
40064001
42004210
40054001
46004210
40054001
45004210
40054001
41104110
40034001
41104110
40034001
55005500
```
INITALIZE SPRITE TABLE

SP0

SP1

SP2

Team A SCORE

SP3

SP4

SP5

Team B SCORE

SP6

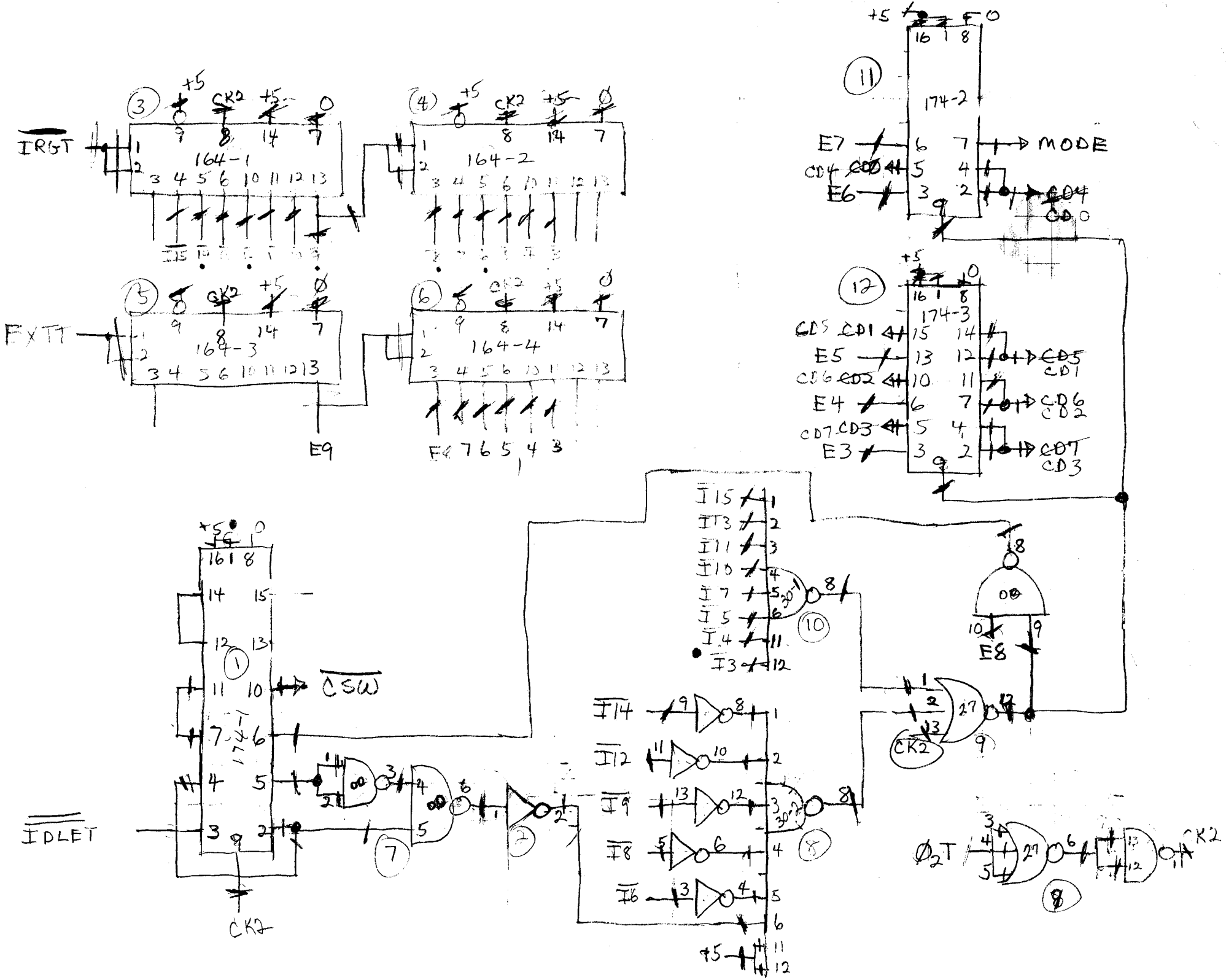
SP7

Point VALUE of Question

SP8

STOP CODE

1000
10


IRGT
164-1
164-2
EXTT
164-3
164-4
E9
E8
174-2
MODE
174-3
CSW
IDLET
CK2
Ø2T

TITLE / TITEL / TITRE: 59 to VDP Software

PAGE / SEITE / PAGE: 1 OF / VON / DE

PROGRAMMER / PROGRAMMIERER / PROGRAMMEUR: Karl M. Guttag

DATE / DATUM / DATE

TI PROGRAMMABLE
CODING FORM
KODEFORM
FEUILLE DE PROGRAMMATION

LOC ADR ADR	CODE KODE CODE	KEY TASTE TOUCHE	COMMENTS BEMERKUNGEN COMMENTAIRES
0		LBL	
1		A'	
2		÷	Divide By
3		1	16 To Get
4		6	Pseudo-Hex
5		=	MS Digit
6		STO	to left of
7		03	decimal
8		8	Put 8 in
9		X≤T	T for Comparison
10		RCL	STRIP off
1		03	Hex MS Digit
2		INT	
3		INV	
4		GE	IF≥8 then
5		−	add 2 to
6		+	Convert to
7		2	Pseudo-Octal
8		LBL	
9		−	
20		+	Add ON
1		4	CONTROL
2		0	Field. 40 gives
3		=	Write to Address
4		X	Register
5		1	Shift MS
6		0	Digit Plus Control
7		0	Right 2 Places
8		=	
9		STO	Save Temp.
30		02	Partial Result
1		RCL	Fetch and
2		03	Strip off MS
3		INV	Hex Digit
4		INT	Then Mult
5		X	By 16 to
6		1	Get LS
7		6	Hex digit
8		=	
9		INV	IF ≥ 8
40		GE	then Add
1		÷	2 to Convert
2		+	to Pseudo-
3		2	Octal.
4		=	
5		LBL	
6		÷	
7		+	Add LS
8		RCL	digit to
9		02	Right Shifted
50		Σ	MS Digit
51		RTN	And Control
2			to get Result
3			
4			

LOC ADR ADR	CODE KODE CODE	KEY TASTE TOUCHE	COMMENTS BEMERKUNGEN COMMENTAIRES
5			
6			
7			
8			
9			
0			
1			
52		LBL	DATA. REG
3		B'	
4		STO	
5		04	
6		A'	TRANSLATE DATA
7		+	add MODE
8		2	TO 1 MASK
9		0	SO that will
60		8	write to VDP
1		8	Address Reg.
2		=	
3		STO	SAVE INTERMED.
4		05	Result
5		RCL	
6		04	STRIP off
7		INV	Reg Number
8		INT	and move
9		X	to Integer
70		1	Place
1		0	
2		=	
3		NOP	
4		+	ADD 7000 TO
5		7	Set WRITE=1
6		0	MODE=1
7		0	and to set
8		0	VDP Reg. Bit
9		=	inside Address
80		X	Register
1		4	X1000
2		INV	TO Move
3		LOG	Reg # & Control
4		+	to left of
5		RCL	DATA & Control
6		05	Then ADD
7		=	The TWO
8		OP	Put Result
9		04	IN PRINT Register
90		0f	"Print" to
1		05	VDP.
2		RTN	
3			
4			
5			
6			
7			
8			
9			

LOC ADR ADR	CODE KODE CODE	KEY TASTE TOUCHE	COMMENTS BEMERKUNGEN COMMENTAIRES
90			
1			
2			
3		LBL	
4		D	
5		2	START TRANSFer
6		8	with Reg 28
7		STO	REG 6 Points
8		06	at Transfer Data'
9		8	Go through
100		STO	LOOP 8 Times
1		05	R5=COUNT
2		LBL	
3		4x	
4		RC*	Fetch Next
5		6	DATA TO TRANSfer
6		OP	Put DATA
7		4	IN PRINT Reg.
8		OP	INC Reg 6
9		26	
110		RC*	Fetch Next
1		06	DATA
2		OP	Put DATA
3		03	IN Print Reg
4		OP	INC
5		26	R6
6		RC*	
7		06	
8		OP	Put DATA
9		02	IN PRINT Reg.
120		OP	
1		26	
2		RC*	
3		06	
4		OP	
5		01	
6		OP	INC Reg
7		26	6
8		OP	"PRINT" OUT
9		05	4 Bytes to VDP
130		DSZ	LOOP Till
1		05	R5=0
2		4*	
3		OP	Clear Print
4		00	Registers
5		RTN	
6			
7			
8			

MERGED CODES
KOMBINATIONS-KODES
TOUCHES COMBINEES

62 Pgm Ind	72 STO Ind	83 GTO Ind
63 Exc Ind	73 RCL Ind	84 Op Ind
64 Prd Ind	74 SUM Ind	92 INV SBR

TEXAS INSTRUMENTS

TITLE
TITEL
TITRE

PROGRAMMER
PROGRAMMIERER
PROGRAMMEUR

PAGE SEITE PAGE ____ OF VON DE ____

DATE DATUM DATE

TI PROGRAMMABLE
CODING FORM
KODEFORM
FEUILLE DE PROGRAMMATION

LOC ADR ADR	CODE KODE CODE	KEY TASTE TOUCHE	COMMENTS BEMERKUNGEN COMMENTAIRES
0			
1			
2			
3			
4			
5			
136		LBL	
7		D'	Get MS
8		÷	BYTE of
9		2	Address to
140		5	Left of
1		6	Decimal Point
2		=	
3		STO	Save to Right
4		01	of Dec. Point
5		INT	TRANLATE
6		A'	MS BYTE
7		X	Move to
8		4	Left 4
9		INV	Decimal Digits
150		LOG	and SAVE
1		=	Partial Result
2		STO	IN REG 7
3		07	
4		RCL	Recall
5		01	Rest of #
6		INV	AND STRIP off
7		INT	MS BYTE.
8		X	Move MS.
9		2	BYTE to
160		5	Right of
1		6	Decimal
2		=	POINT
3		A'	TRANSLATE
4		+	LS BYTE
5		2	ADD ON
6		4	Control
7		0	Signals
8		0	
9		2	
170		0	
1		0	
2		0	
3		=	
4		+	ADD TO
5		RCL	TRANSLATED
6		07	MS BYTE
7		=	
8		OP	PUT IN
9		04	PRINT Reg.
180		OP	"PRINT"
1		05	TO VDP.
2		RTN	
3			
4			

LOC ADR ADR	CODE KODE CODE	KEY TASTE TOUCHE	COMMENTS BEMERKUNGEN COMMENTAIRES
5			
6			
7			
8			
9			
0			
1			
2			SAVE TRANS.
1803		LBL	DATA AT
4		E'	R3
5		ST*	
6		03	
7		OP	
8		23	
9		RTN	
0			
1			
2			
3			
4			
5			
6			
7			
8			
9			
190		LBC	
1		C'	
2		STO	
3		05	
4		4	Note that
5		0	"40" Represents
6		4	a WRITE
7		0	DATA=0 if
8		4	the 59 to VDP
9		0	transfer latches
200		4	Start out
1		0	Cleared due
2		0	to the starting
3		0	4000 Code
4		OP	STORE
5		01	"CLEAR four"
6		OP	IN Four
7		02	Print Registers
8		OP	So that get
9		03	16 memory
0		OP	Locations Clear
1		64	for L200P
2		LBL	
3		EE	
4		OP	"Print" Clear
5		05	of 16.
6		DSZ	
7		EE	
8		05	
9		RTN	

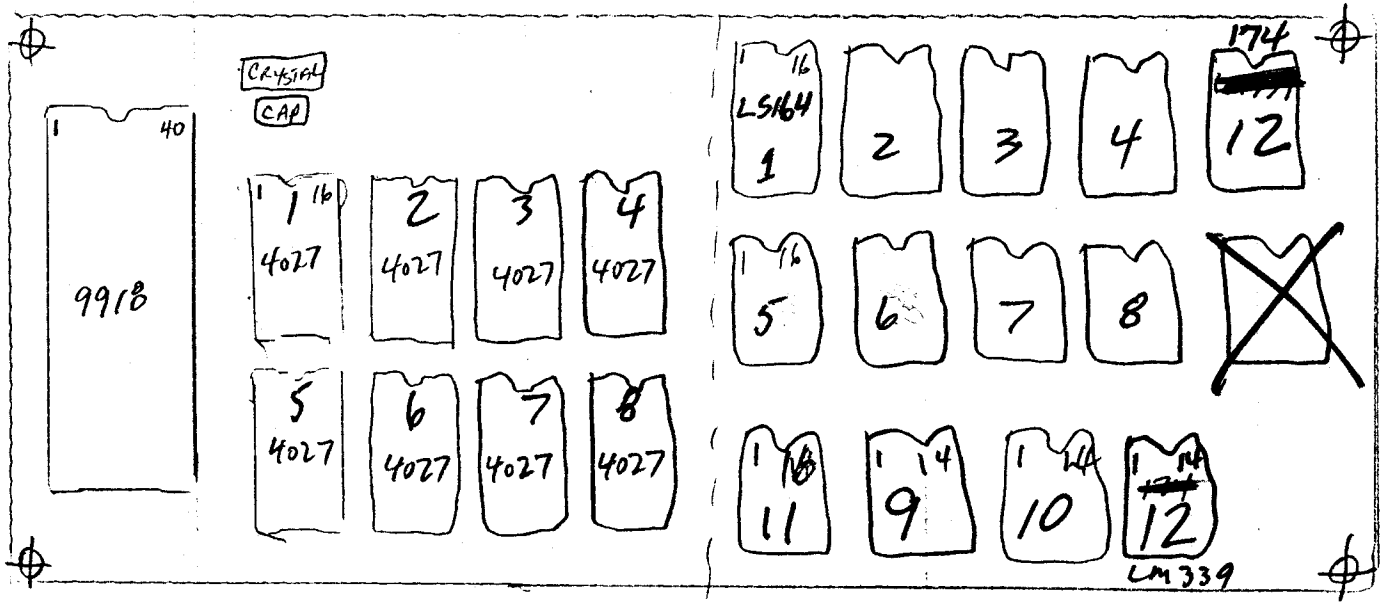
LOC ADR ADR	CODE KODE CODE	KEY TASTE TOUCHE	COMMENTS BEMERKUNGEN COMMENTAIRES
0			
1			
2			
3			
4			
5			
6			
7			
8			
9			
0			
1			
2			
3			
4			
5			
6			
7			
8			
9			
0			
1			
2			
3			
4			
5			
6			
7			
8			
9			
0			
1			
2			
3			
4			
5			
6			
7			
8			

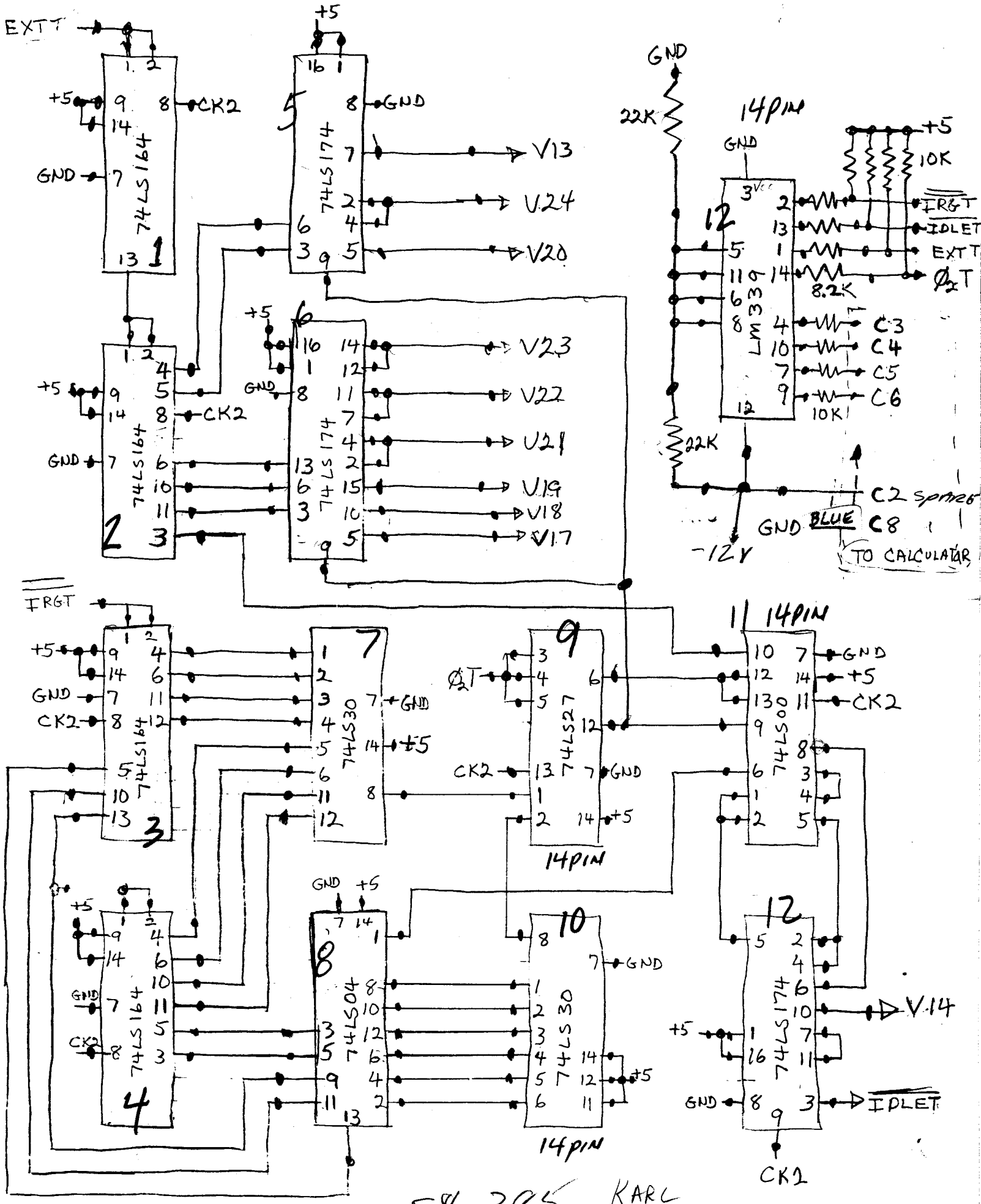
MERGED CODES
KOMBINATIONS-KODES
TOUCHES COMBINEES

62 Prm Ind		72 STO Ind		83 GTO Ind	
63 Fix Ind		73 RCL Ind		84 Dsz Ind	
64 Prd Ind		74 SUM Ind		92 INV SBR	

TEXAS INSTRUMENTS

LOGIC BOARD


1
40
9918
CRYSTAL
CAP
1 16
4027
1
2
4027
3
4027
4
4027
5
4027
6
4027
7
4027
8
4027
1 16
LS164
1
2
3
4
174
12
1 16
5
6
7
8
1 16
11
1 14
9
1 14
10
1 14
12
LM339


EXTT
+5
CK2
GND
74LS164
74LS174
V13
V24
V20
V23
V22
V21
V19
V18
V17
22K
14PIN
10K
IRGT
IDLET
EXTT
Ø2T
8.2K
LM339
C3
C4
C5
C6
C2 speed
GND
BLUE
C8
TO CALCULATOR
-12V
74LS30
74LS27
74LS00
74LS04
V14
IDLET
CK2
1
2
3
4
5
6
7
8
9
10
11
12
KARL
JOE