ABSTRACT

An interface has been designed to enable any device with a 300 baud RS-232 serial port to communicate with the commercially available HORNBY Zero-1 model train controller. One can send commands through the interface to the Hornby device which perform the same function as depressing a button on the keypad or manipulating the slide switch on the front of the unit. Since most microcomputers have a serial port, they can be connected to the interface and can be programmed to perform many of the functions that would have been done painstakingly with the Hornbay alone; hence, the interface has enabled a much more "user friendly" environment.

INTRODUCTION

To attempt to control sixteen locomotives at once with the Hornby unit alone is extremely difficult and burdensome. Although the company offers "slave" modules which can give the user real time control over four locomotives at once, and an extension is available which extends to sixteen separate slide controls, one for each loco, it is very costly to do so, and really does not take full advantage of the microprocessor controlled Hornby unit. By enabling any device with a 300 baud RS-232 port to communicate with the Hornby, we can use the Hornby much effectively.

Originally, I set out to improve the user friendliness of the Hornby unit. As mentioned above, simultaneous control of many locomotives is very cumbersome, if not just because of the number of keystrokes necessary to change control from one locomotive to another. By attaching a microcomputer to the Hornby, the method of user interaction is limited only by the speed at which the Hornby can detect keystrokes the

maximum communication rate through the interface--about 10 characters per second) and the imagination of the programmer. I have written a set of subroutines in MBASIC (microsoft, inc.) which make the method of communication transparent to the user; he need only know which parameters are necessary to control any given locomotive or send messages to an accessory. It is quite simple for even a novice programmer to use these routines to achieve a much more efficient and friendly interface to the Hornby unit and his model train layout.

I have included three sample programs which use the general subroutines; one enables the user to control any one of ten trains at once (speed, direction), another sends messages to the accessories, and the third combines these two to demonstrate the "user friendly" nature of this interface configuration. The third program takes advantage of the many keys of the computer keyboard and gives the user real time control over six trains while also enabling him to send messages to the accessories. Also, an important aspect of any user interface is introduced: feedback--a visual display of the respective speeds of the locomotives and the message pending or sent to the accessories.

The initial philosophy that I adopted was to let the Hornby continue to do the microprocessor control; the interface would only communicate with the Hornby in some way without attempting any direct locomotive or accessory con-

trol. The interface does not conflict with the normal operation of the Hornby; it is also easily installed without having to "cut traces" or really disturb the existing controller much. The circuit designed can be easily put onto a printed circuit board and installed inobtrusively underneath the Hornby unit.

The interface that has been designed should be rather inexpensive to produce, especially compared with the alternative purchase of sixteen slave units. But, more importantly, the interface enables the user to take full advantage of the power of a separate microcomputer and the inherent power of the Hornby.

THEORY OF OPERATION

HARDWARE

General

Refer to the appendices for schematics and drawings. The interface operates by receiving control characters via a standard, non hand-shaked, 300 baud RS-232 line, decoding the characters, and acting appropriately. All commands to the interface consist of a single legal control character (see figures 1 and 2) corresponding to a single action on the Hornby (e.g. depressing the 'LOCO' key on the Hornby is equivalent to sending the character '<' to the interface when the interface is enabled). All illegal characters are

decoded out and do not effect the interface (although some dummy characters must be sent to separate each legal control character--see keyboard control below).

The hardware does not interfere with the operation of the Hornby; rather, it simulates the keyboard switch closures, or it couples the appropriate signal to the TMS1000 microcomputer within the Hornby according to the character sent to the interface by the user (see figures 1 and 2 for character codes). Most of the interface consists of random logic CMOS integrated circuits, powered by the Hornby's internal 15 volt power supply (a higher power 7815 voltage regulator is installed on the Hornby PC board instead of the existing 78L15); the remainder of the interface circuitry is dedicated to the communications interface--an RS-232 line receiver, a baud rate generator, and a UART--powered by a 78L05 regulator also attached to the 15 volt supply.

It seems that, where there would normally be "contact closures" (e.g. keypad switches), a capacitive coupling is used by the Hornby. The interface employs CMOS 4066 analog switches to simulate these closures. The TMS1000 has been used to control a microwave oven with a capacitive keypad; this is probably why the Hornby uses a capacitive coupling system. The TMS1000 accepts the 4066 closure as equivalent to the capacitive couple without any trouble.

Speed	Control Character	ASCII Code
0	"A"	100 0001
1	"C"	100 0011
2	"B"	100 0010
3	"F"	100 0110
4	"G"	100 0111
5	"E"	100 0101
6	"D"	100 0100
7	"L"	100 1100
8	"M"	100 1101
9	"O"	100 1111
10	"N"	100 1110
11	"J"	100 1010
12	"K"	100 1011
13	"I"	100 1001
14	"H"	100 1000

Figure 1 - Speed Control Characters

Hornby Key	Control Character	ASCII Code
Ø	"Ø"	011 0000
1	"1"	011 0001
2	"2"	011 0010
3	"3"	011 0011
4	"4"	011 0100
5	"5"	011 0101
6	"6"	011 0110
7	"7"	011 0111
8	"8"	011 1000
9	"9"	011 1001
INERTIA	":"	011 1010
REVERSE	";"	011 1011
LOCO	"<"	011 1100
←	"="	011 1101
→	">"	011 1110
FORWARD	"?"	011 1111

Figure 2 - Keyboard Control Characters

Speed control

For normal operation, the user controls the speed of the locomotives with the slide switch on the front panel of the Hornby. The slide switch selectively couples together a common signal (Css) to one or more of the four inputs to the TMS1000 microcomputer within the Hornby (K1, K2, K4, and K8--not equivalent to the keyboard lines labelled identically. I label these lines K1ss, K2ss... on my schematics to differentiate them from the keyboard lines which I label K1kbd, K2kbd...). When using the interface, the Hornby's front panel slide switch is disabled; the interface is then free to simulate the switch's encoding by selectively coupling one or more of the Css signal to the four Kss lines with a CMOS 4066 analog switch.

The UART receives a speed control character (see figure 2 for codes) and decodes it for a bank of four analog switches (one 4066) to couple the Css signal to the appropriate Kss lines.

Direction control

One controls the direction of the locomotives by pressing either the forward or reverse switch on the front panel of the Hornby. Internally, this just couples or uncouples a signal from one of the two lines labelled "SW" to the other. The interface uses two control characters to determine the direction setting. A "?" is interpreted as a command go

forward, a "·" to go in reverse. Once a direction control is decoded, the interface sets an RS flip-flop (two cross-coupled NAND gates) which either closes or opens a 4066 switch across the two "SW" lines

Keyboard control

In normal operation, the Hornby's keypad is encoded to couple one of the five lines labelled R5 (not used by the interface), R6, R7, R8, and R9 to one of the lines labelled K1, K2, K4, and K8 on the keypad (I label them K1kbd, K2kbd... because these are not equivalent to the lines labelled the same for the slide switch). The interface simply decodes the control character sent to it by the user, and couples the appropriate signal to the Kkbd lines for a period of time determined by a one-shot (50mS). This one-shot operation is necessary to simulate the "key-push" of the user on the normal keypad. Since the TMS1000 scans the keypad assuming that a human is depressing the keys, the rate of key depression is limited (hence the 50mS key press). Also, the TMS1000 expects to see no key pressed for a time after a key press; at 300 baud, this means that two dummy (blank) characters must be sent after each legal control character. For this reason, a legal control character is latched until the next legal one arrives. The UART can be configured to receive characters at 1200 baud, but this will not speed the rate of communication between the user and the

Hornby. At 1200 baud, eleven dummy characters must be sent between each control character. All of this means that we are simulating a key press of 50mS, and a key release of 49mS (three 300 baud, or twelve 1200 baud characters=99mS). Hence, we actually have an effective communication rate with the Hornby of 100 baud (ten characters per second).

PRACTICAL OPERATION

To use the interface, simply connect a cable between the RS-232 port of the device you are using to the DB-25 plug on the back of the Hornby. Pin 7 is the signal ground, and pin 2 is the signal (the interface is treated exactly as a line printer). To use the Hornby without the interface, put the "Hornby/interface" switch in the Hornby position--the interface will be locked out (no characters will be accepted). To use the interface, just put the switch in the interface position. In this mode, the Hornby keypad will continue to function (except the reverse and forward buttons) so that you can still hit the 'PANIC' or 'CLEAR' keys (or any others if you want); but, the speed control slide switch will have no effect while the switch is in the interface position.

One need not be concerned with the complexities of the Hornby nor with the intricacies of the interface to take advantage of this new tool. Its operation is quite straightforward; in fact, a computer is not even necessary to use the interface. Any device with a 300 baud RS-232 port will

work to control the Hornby with the interface enabled. Using the control characters given in figures 1 and 2, one can simulate a key press on the Hornby, or set the speed for a locomotive (NOTE: it is wise to insert dummy blanks between each control character, even when just typing the characters into the interface). For example, one could type ": 3 < 5 = D ;" which would mean to set locomotive number 5 to inertia 3, speed 8, in reverse; this is equivalent to pressing the keys 'INERTIA' '3' 'LOCO' '5' '<--', pressing the reverse switch, and setting the speed control to speed 8.

So far, this does not seem like much of an accomplishment; but, when we attach the power of a microcomputer to the interface, we can program entire scenarios which will be sent over the RS-232 port much faster than we can even think. Or, the programmer can set up the computer's keys to correspond to various functions such as raising or lowering the speed of any of a few locomotives (see example program 3). If the user has a real time clock, he could set up a scenario of events which the computer could initiate at the correct time by sending the appropriate characters to the Hornby.

By using the statement LPRINT ": 3 < 5 =", the user would have control over locomotive 5, and the inertia will be set to 3. The user can then use the statement LPRINT "?" to set the direction to forward and the statement LPRINT "C" to start the locomotive at the lowest speed. One must fol-

low all conventions of the Hornby, of course; if one were to follow the previous statements with LPRINT "< 1 > G", locomotive five would continue as set before, and locomotive one would go to speed 6, in the same direction as loco 5 (forward). Any further speed commands would not affect loco 5, but would control loco 1, just as if we had pressed the keys on the Hornby itself.

SOFTWARE

GENERAL SUBROUTINES -- LOCOMOTIVE CONTROL AND ACCESSORY MESSAGE SENDING

A general interface subroutine section and three example programs which use these subroutines are provided with this report. A program is also provided for the Osborne 1 which is specific to this microcomputer. The general routines make the method of communication with the Hornby transparent to the user; he need only keep track of the appropriate parameters and call the necessary subroutines which take care of all of the ugly work.

NOTE 1: To change the baud rate to 1200 baud, the variable on line number 37 must be set to nine blanks. This should only be necessary if your RS-232 device has no 300 baud setting. You must also alter the hardware to do so, though, by resetting the baud rate generator select lines.

NOTE 2: Some of the code is language dependent. It is all MBASIC, an extremely standard language, but many of te

computers available do not have it (e.g. Apple with its equivalent Applesoft). The statements LPRINT, INSTR, and MID$ might differ from BASIC to BASIC; hence, these routines are meant as examples of how to use the interface, but are not really intended as "end user" equipment. The routines are written in such a manner, though, that one can just adapt them to their computer in MBASIC; for example, one can change the output routine to suit their fancy in EXAMPLE PROGRAM 3 according to the capability of their specific computer.

The first routine is the initialization section. This must be called by stating GOSUB 5 at the beginning of any program which is to use these general routines.

The second of these routines accepts the parameters LOCO (the number of the locomotive to be controlled), SPD (the speed - negative speeds mean reverse, positive mean forward), and INERT (the inertia to be set). The values of LOCO can range from 1 to 16, SPD from -14 to 14, and INERT from 0 to 4 (0 means don't bother to fool with the inertia). To set loco 6 to speed 3, in reverse, the following code would be appropriate:

```
1000 GOSUB 5
1010 LOCO = 6
1020 SPD = -3
1030 INERT = 0
1040 GOSUB 100
```

The subroutine will not send out unnecessary characters, though; it keeps track of the present inertia, direction,

and locomotive. For example, it is not necessary to hit the key sequence 'LOCO' '5' '=' to set the speed of loco 5 with a 'G' if you are already controlling loco 5. Similarly it is not necessary to send out the direction control character if the direction is not to change, even if you are going to control a different locomotive. Also, carriage returns aren't unnecessarily printed out; they are used sparingly, so as to make the interface respond as fast as possible. Of course, the user need not concern himself with this, unless he wishes to write a similar routine for himself.

The third, and last routine is used to send messages to the accessories. The parameters NUM1 and NUM2 are the two numbers to be sent, and ENTR determines whether a left arrow or right arrow is used to enter the numbers (ENTR=0 for left, 1 for right).

EXAMPLE PROGRAM 1

To run this program you must type LOAD "HORNEX1.BAS" and then RUN 1000 (because the program itself begins at 1000--the general subroutines are between 1 and 999). This program prompts the user for the necessary information and uses the general routines to control any one locomotive at a time. The routine also keeps track of the individual locomotive speeds and prints them out after each control command. This is a very simple demonstration of the use of the locomotive control routines.

EXAMPLE PROGRAM 2

Type LOAD "HORNEX2.BAS" and RUN 1000 to use this program. This example demonstrates the method of sending messages to the accessories using the general subroutines.

EXAMPLE PROGRAM 3

Type LOAD "HORNEX3.BAS" and RUN 1000 to run this example. The most complicated of the three examples, this program combines the first two and adds a little in the way of a nicer display. An explanation of the use of this program is really necessary, though. The keyboard functions as a control over locomotives 1 through 6. Loco 1 is controlled by the keys Q, A, and Z, loco 2 with W, S, and X, and so on for the other four locos (uppercase characers only). The upper of the three keys on the keyboard for each loco accelerates the appropriate loco one speed, the middle stops the loco, and the bottom decelerates the loco one speed. For example, if you were to press "RRRZYF", you would accelerate loco 4 to speed 3, set loco 1 to speed 1 in reverse, set loco 6 to speed 1, and then stop loco 4; you must of course wait for the computer to respond or it might miss some of the characters. To send a message, you merely have to type the number keys corresponding to the message you wish to send (e.g. "74") and then either the "-" key or the "=" key for the left or right arrow, respectively. If you make a mistake with the message before you hit the respective arrow key,

you can just keep entering the numbers until you are satisfied. For example, if you were to enter "973474-" in sequence, the message "7 4 <--" would be sent through the Hornby to the accessory. The display also tells you the present speed of all of the locomotives and the present message sent or pending (there will be an appropriate arrow following the message in parenthesees if it has already been sent).

EXAMPLE FOR THE OSBORNE 1

This was written before the general subroutines, hence the somewhat sloppy code. The program can be run by typing RUN "HORN1.BAS." It is very Osborne specific and is not intended for any other system. It operates exactly as Example 3, but the control is extended for nine locomotives ("O", "L", and "." control loco 9, for example -- see EXAMPLE PROGRAM 3 above). The main difference is the elegance of the display, though.

CONCLUSION

At the outset, this project was an attempt to improve the user friendliness of the Hornby Zero-1 model train controller; it quickly developed into a research project dealing with the workings of the Hornby unit -- much time was spent in making a reliable, compact, general, and efficient interface. The interface achieves all of these goals and enables

the ambitious programmer to take full advantage of the Hornby controller. I was pleased to demonstrate to myself, as well as others, the improvements that can be made in the user interface with the Hornby, as well as the ease with which these improvements can be made, even with the simplest of programs written in MBASIC. The interface should be of interest to both computer and model train enthusiasts alike. It draws the computer owner to the world of the model train by enabling him to exercise real time control with the powerful Hornby unit, and it awakens the model railroader to the powerful world of the computer.

```
REM ************************************************
REM *** HORNBY/RS-232 INTERFACE SUBROUTINES ***
REM ************************************************
REM COPYRIGHT 1/83 PHIL DWORSKY
REM *** INITIALIZATION SUBROUTINE ***
DIM INERTCH$(1),RARRCH$(1),LARRCH$(1),FORWRDCH$(1),REVERSCH$(1),LOCOCH$(1)
DIM SPEED$(10)
 INERTCH$="!"
 RARRCH$=">"
 LARRCH$="="
 FORWRDCH$="T"
 REVERSCH$="}"
 LOCOCH$="<"
 SPEED$="ACBFGEDLMONJKIH"
 REM **FILL=0 BLANKS FOR 300 BAUD, 9 BLANKS FOR 1200 BAUD ***
 FILL$=""
 PRESLOCO=0
 PRESINERT=0
 PRESDIR=1234
 INERT=0
 LOCO=0
 SPD=0
 RETURN
0 REM ************************************************
0 REM *** SET LOCOMOTIVE (LOCO) TO VELOCITY (SPD), INERTIA (INERT) ***
1 REM ************************************************
0 IF (LOCO<1) OR (LOCO>16) OR (ABS(SPD)>14) OR (INERT<0) OR (INERT>4) THEN
0 IF (LOCO<>PRESLOCO) OR ((INERT<>PRESINERT) AND (INERT<>0)) THEN GOSUB 200
0 IF (PRESDIR=SGN(SPD)) OR (SPD=0) THEN 180
0 PRESDIR = SGN(SPD)
0 IF SPD>0 THEN 180
0 LPRINT REVERSCH$;"  ";FILL$;
0 GOTO 185
0 LPRINT FORWRDCH$;"  ";FILL$;
0 LPRINT MID$(SPEED$,1+ABS(SPD),1);"  ";FILL$
0 RETURN
0 REM *** NEW LOCO OR INERTIA... ***
0 IF ((INERT<>PRESINERT) OR (LOCO<>PRESLOCO)) AND (INERT<>0) THEN LPRINT
0 LPRINT LOCOCH$;" ";FILL$;
0 IF LOCO>9 THEN 260
0 LPRINT LOCO;FILL$;
0 GOTO 270
0 LPRINT " 1 ";FILL$;LOCO-10;FILL$;
0 LPRINT " ";LARRCH$;"  ";FILL$;
0 PRESINERT=INERT
0 PRESLOCO=LOCO
0 RETURN
99 REM ************************************************
00 REM *** SEND A MESSAGE(NUM1,NUM2--EACH 0 TO 9) WITH ENTER KEY ***
01 REM *** DETERMINED BY ENTR (0=LEFT ARROW, NOT 0-RIGHT)          ***
02 REM ************************************************
05 IF (NUM1<0) OR (NUM1>9) OR (NUM2<0) OR (NUM2>9) THEN RETURN
10 LPRINT NUM1;FILL$;NUM2;" ";FILL$;
20 IF ENTR <> 0 THEN LPRINT RARRCH$;"  ";FILL$
30 IF ENTR = 0 THEN LPRINT LARRCH$;"  ";FILL$
40 RETURN
```

(Line 210 continued) → " ";FILL$;INE
F

```
1000 REM *** EXAMPLE PROGRAM #1 ***
1001 REM *** CONTROL ONE LOCOMOTIVE AT A TIME ***
1002 REM COPYRIGHT 1/83 PHIL DWORSKY
1003 GOSUB 5
1004 DIM SPDTABLE(16)
1005 FOR I = 1 TO 16
1006 SPDTABLE(I)=0
1007 NEXT I
1010 INPUT "Which locomotive to control (0 to QUIT)";LOCO
1015 IF LOCO=0 THEN 1060
1020 INPUT "What speed (-14 to 14)";SPD
1025 INPUT "What inertia (0=same, 1 thru 4 to set)";INERT
1030 GOSUB 100
1040 SPDTABLE(LOCO)=SPD
1045 GOSUB 1090
1050 GOTO 1010
1060 REM *** QUIT ***
1070 PRINT "Done."
1080 END
1090 REM *** PRINT OUT RESPECTIVE SPEEDS ***
1091 FOR I = 1 TO 16
1092 PRINT SPDTABLE(I);
1100 NEXT I
1105 PRINT
1110 RETURN
```

```
1000 REM *** EXAMPLE PROGRAM #2 ***
1001 REM *** ACCESSORY CONTROL ***
1002 REM COPYRIGHT 1/83 PHIL DWORSKY
1003 GOSUB 5
1010 PRINT "Input the two digits separated by a"
1011 INPUT "comma (0,0 to quit)";NUM1,NUM2
1012 IF (NUM1=0) AND (NUM2=0) THEN 1080
1020 INPUT "Left arrow (0) or right (1)";ENTR
1030 GOSUB 400
1040 PRINT "Message: ";NUM1;" ";NUM2;"  ";
1050 IF ENTR <> 0 THEN PRINT "--> Sent."
1060 IF ENTR = 0 THEN PRINT "<-- Sent."
1070 GOTO 1010
1080 REM *** QUIT ***
1090 PRINT "Done."
1100 END
```

```
1000 REM ************************************************************
1001 REM *** EXAMPLE PROGRAM #3                                     ***
1002 REM *** CONTROL LOCO'S 1 THRU 6 AND SEND MESSAGES             ***
1003 REM ************************************************************
1004 REM COPYRIGHT 1/83 PHIL DWORSKY
1005 GOSUB 5
1010 DIM SPDTABLE(6)
1015 FOR I = 1 TO 6
1020 SPDTABLE(I)=0
1025 NEXT I
1030 TRUE=-1
1035 FALSE=0
1040 SENT = FALSE
1045 KBD$="QAZWSXEDCRFVTGBYHN"
1050 MESG=0
1055 DIM IN$(1)
1060 KBDLARR$="-"
1065 KBDRARR$="="
1100 REM OUTPUT STATUS TO SCREEN
1110 GOSUB 2000
1120 REM GET CHARACTER (IN$) FROM TERMINAL
1130 GOSUB 2100
1140 REM DETERMINE IF SPEED OR MESSAGE CHARACTER
1150 IF (VAL(IN$)=0) AND IN$<>"0" THEN 1210
1160 REM A NUMBER--MESSAGE-ONLY LAST TWO ENTERED
1170 IF SENT THEN MESG = 0
1180 MESG = 10*(MESG-10*(INT(MESG/10)))+VAL(IN$)
1190 SENT = FALSE
1200 GOTO 1100
1210 IF (IN$=KBDLARR$) OR (IN$=KBDRARR$) THEN 1300
1220 REM SPEED CONTROL CHARACTER ENTERED
1225 REM FIGURE OUT WHICH LOCO, WHICH FUNCTION (ACCEL,DECEL,STOP)
1230 KEYNUM=INSTR(KBD$,IN$)
1240 IF KEYNUM=0 THEN 1100
1250 LOCO=1+INT((KEYNUM-1)/3)
1255 IF LOCO=0 THEN 1100
1260 FUNC = 1+((KEYNUM-1)MOD 3)
1270 ON FUNC GOTO 1280,1310,1340
1280 REM INCREASE SPEED OF LOCO
1290 SPD=SPDTABLE(LOCO)+1
1300 GOTO 1360
1310 REM STOP LOCO
1320 SPD = 0
1330 GOTO 1360
1340 REM DECREASE SPEED OF LOCO
1350 SPD = SPDTABLE(LOCO)-1
1360 REM REFLECT IN TABLE AND SEND TO HORNBY
1370 IF SPD<-14 THEN SPD = -14
1380 IF SPD>14 THEN SPD = 14
1390 SPDTABLE (LOCO) = SPD
1400 GOSUB 100
1410 GOTO 1100
1500 REM ** SEND COMPLETE MESSAGE **
1510 SENT = TRUE
1520 NUM1=INT(MESG/10)
1530 NUM2=MESG-10*NUM1
1540 IF IN$=KBDLARR$ THEN ENTR = 0
1550 IF IN$=KBDRARR$ THEN ENTR = 1
1560 GOSUB 400
1570 GOTO 1100
2000 REM *** OUTPUT STATUS AND MESSAGE TO SCREEN ***
```

Example Program 3 page 3

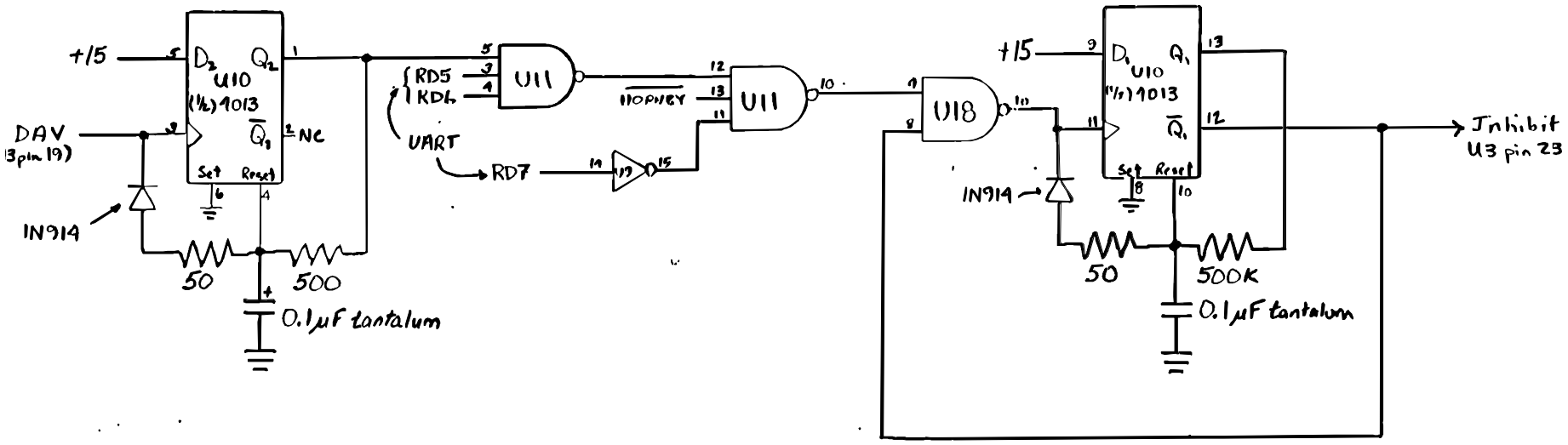
```
2010 FOR I = 1 TO 8
2020 PRINT SPDTABLE(I);
2025 IF ABS(SPDTABLE(I))<10 THEN PRINT " ";
2030 NEXT I
2040 DIG1 = INT(MESG/10)
2045 DIG2 = MESG-10*DIG1
2050 PRINT " (";DIG1;DIG2;
2060 IF NOT SENT THEN 2090
2070 IF ENTR = 0 THEN PRINT "<-- )";
2080 IF ENTR <> 0 THEN PRINT "--> )";
2085 RETURN
2090 PRINT "    )";
2095 RETURN
2100 REM *** INPUT A CHARACTER (IN$) SUBROUTINE
2105 REM ADAPT THIS ROUTINE TO SUIT YOUR PREFERENCE
2110 IN$=INKEY$
2120 IF LEN(IN$) = 0 THEN 2110
2130 RETURN
```

OSBORNE EXAMPL

```
1 GOTO 100
10 REM PHIL DWORSKY  12/14/82  EECS 497  COPYRIGHT 1/83
20 REM HORNBY CONTROL PROGRAM                P.DWORSKY
30 REM CONTROLS NINE TRAINS AT ONCE
35 REM THE TRAINS THAT ARE CONTROLLED ARE IN THE
36 REM ARRAY CALLED TRNS...
40 REM USE THE VERTICAL TRIPLETS
50 REM OF CHARACTERS TO CONTROL EACH TRAIN
60 REM E.G. Q TO RAISE THE SPEED OF TRAIN 1
70 REM Z TO LOWER IT; AND A TO STOP IT.
97 REM
98 REM **** CURSOR CONTROL CHARACTERS ****
99 REM
100 LFT$=CHR$(8)
110 RT$=CHR$(12)
120 UP$=CHR$(11)
130 DOWN$=CHR$(10)
140 HOM$=CHR$(26)
150 LITE$=CHR$(27)+"("
160 DARK$=CHR$(27)+")"
1000 REM *** DECLARATIONS ***
1010 PRESTRN%=0
1015 DIM TRN%(9)
1020 FOR XX=1 TO 9: TRN%(XX)=XX:NEXT XX
1030 LARR$="<  ":KBDLARR$="-"
1040 RARR$=">  ":KBDRARR$="_"
1060 KBD$="QAZWSXEDCRFVTGBYHNUJMIK,OL."
1070 MSG$=""
1080 DIM TRSPD$(9)
1081 FOR XX=1 TO 9:TRSPD$(XX)=0:NEXT XX
1089 DIM SPEED$(10)
1090 SPEED$="ACBFGEDLMONJKIH"
1100 LOCO$="( "
1110 INERTIA$=": "
1120 FORWARD$="?"
1130 REVERSE$=";"
1140 MAXSPD%=14
1150 TRUE=-1
1160 FALSE=0
1170 SENT=FALSE
1180 RETCHR$=CHR$(13)
1500 REM *** SET UP THE GRAPHICS ON THE SCREEN ***
1510 PRINT HOM$
1515 PRINT "      LITTMAN/DWORSKY  Multi-Train Control"
1516 PRINT:PRINT:PRINT
1520 PRINT "   TRAIN#   1   2   3   4   5   6   7   8   9"
1530 PRINT "   ------  --- --- --- --- --- --- --- --- ---"
1998 REM
1999 REM
2000 REM *** GET TERMINAL INPUT ***
2001 PRINT "   Speed:  ";
2002 PRINT DARK$;
2003 FOR XX=1 TO 9 :PRINT TRSPD$(XX);" ";:NEXT XX
2004 PRINT:PRINT UP$;
2005 PRINT LITE$;
2010 IN$=INKEY$
2011 IF LEN(IN$)=0 THEN 2010
2020 IF (VAL(IN$)=0) AND IN$ <> "0" THEN 2100
2029 REM
2030 REM *** A NUMBER HAS BEEN INPUT ***
2031 REM
2035 IF SENT THEN 2040
2036 MSG$=MSG$+IN$+"  "
2037 IF LEN(MSG$)<4 THEN MSG$=RIGHT$(MSG$,4)
```

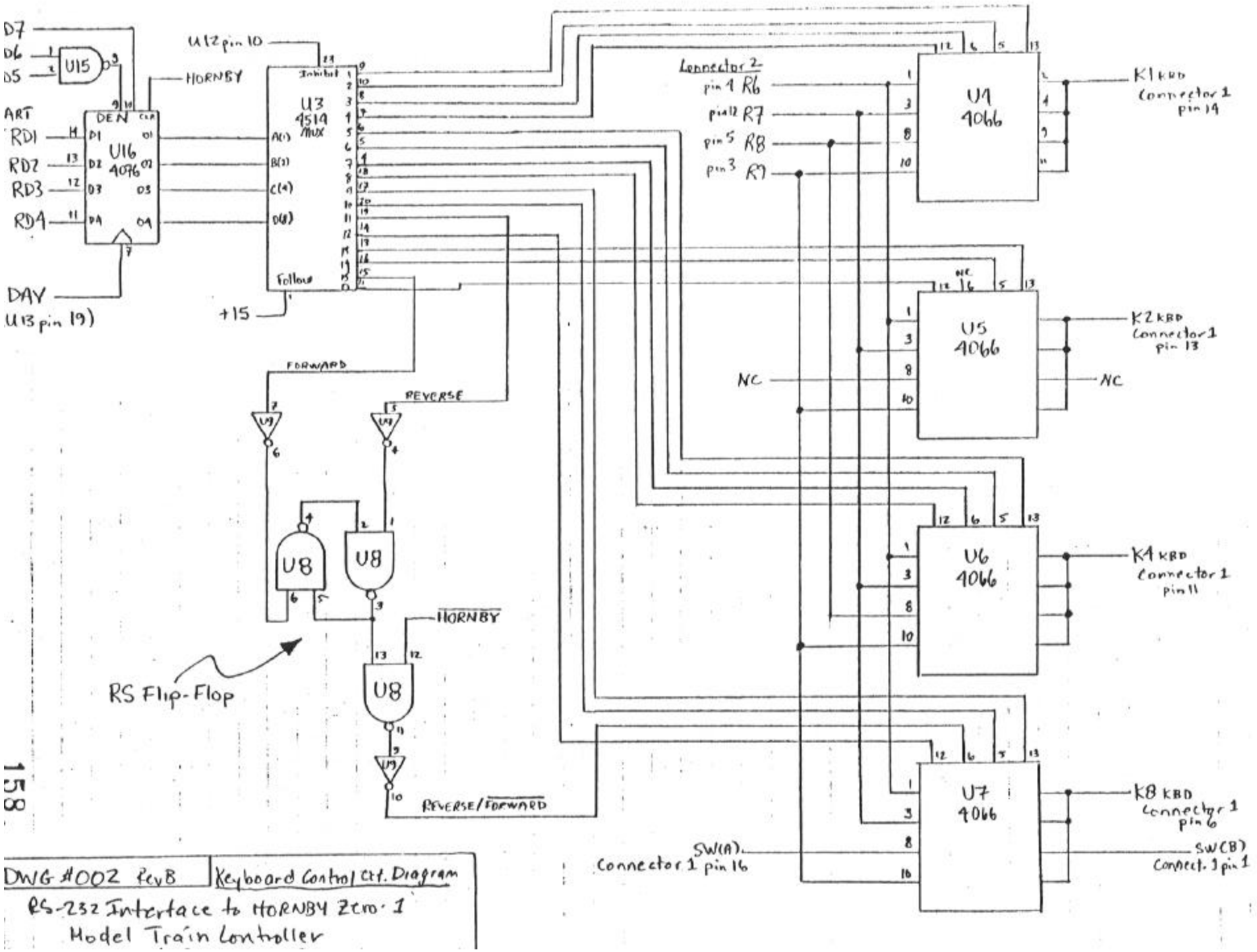
OSBORNE EXAMPLE pag

```
2040 SENT=FALSE
2041 MSG$=IN$+"  "
2045 IF LEN(MSG$)>6 THEN MSG$=RIGHT$(MSG$,3)
2046 FOR XX=1 TO 6:PRINT DOWN$;:NEXT XX
2047 PRINT "              Message: ";DARK$;MSG$;LITE$;"        "
2048 FOR XX=1 TO 7 : PRINT UP$;:NEXT XX
2050 GOTO 2000
2100 IF (IN$=KBDLARR$) OR (IN$=KBDRARR$) THEN 3000
2105 IF IN$=RETCHR$ THEN 3000
2109 REM
2110 REM *** A KEYBOARD CHARACTER--TRAIN CONTROL ***
2120 KEYNUM%=INSTR(KBD$,IN$)
2124 IF KEYNUM% = 0 THEN 2000
2125 IND%=1+INT((KEYNUM%-1)/3)
2127 FUNC% = 1 + ((KEYNUM%-1) MOD 3)
2130 IF IND%=0 THEN 2000
2140 TRINPUT%=TRN$%(IND%)
2150 IF TRINPUT% = PRESTRN% THEN 2200
2154 REM
2155 REM *** A NEW TRAIN TO CONTROL ***
2160 PRESTRN%=TRINPUT%
2165 LPRINT LOCO$;PRESTRN%;" ";LARR$;"  ";
2167 LPRINT MID$(SPEED$,1+ABS(TRSPD$%(PRESTRN%)),1);"  ";
2170 IF TRSPD$%(PRESTRN%)>=0 THEN LPRINT FORWARD$
2180 IF TRSPD$%(PRESTRN%)<0 THEN LPRINT REVERSE$
2199 REM
2200 REM *** PROCESS THE CHARACER ***
2201 ON FUNC% GOTO 2210,2400,2300
2210 REM *** INCREASE SPEED OF PRESTRN% ***
2220 NEWSPD%=TRSPD$%(PRESTRN%)+1
2230 IF NEWSPD%>MAXSPD% THEN 2000
2240 TRSPD$%(PRESTRN%)=NEWSPD%
2250 IF NEWSPD%=1 THEN LPRINT FORWARD$;
2260 LPRINT MID$(SPEED$,1+ABS(NEWSPD%),1);"  "
2270 GOTO 2000
2299 REM
2300 REM *** DECREASE SPEED OF PRESTRN% ***
2310 NEWSPD%=TRSPD$%(PRESTRN%)-1
2320 IF ABS(NEWSPD%)>MAXSPD% THEN 2000
2330 TRSPD$%(PRESTRN%)=NEWSPD%
2340 IF NEWSPD%=-1 THEN LPRINT REVERSE$;
2350 LPRINT MID$(SPEED$,1+ABS(NEWSPD%),1);"  "
2360 GOTO 2000
2399 REM
2400 REM *** STOP PRESTRN% ***
2410 LPRINT MID$(SPEED$,1,1);"  "
2420 TRSPD$%(PRESTRN%) = 0
2430 GOTO 2000
2999 REM
3000 REM *** SEND COMPLETE MESSAGE TO ACCESSORY ***
3005 IF LEN(MSG$)<6 THEN 2000
3008 SENT=TRUE
3010 LPRINT MSG$;
3020 IF IN$=KBDLARR$ THEN LPRINT LARR$;
3030 IF IN$=KBDRARR$ THEN LPRINT RARR$;
3040 FOR XX=1 TO 6 :PRINT DOWN$;:NEXT XX
3050 PRINT "              Message: ";MSG$;
3060 IF IN$=KBDLARR$ THEN PRINT " <--"
3070 IF IN$=KBDRARR$ THEN PRINT " -->"
3080 FOR XX=1 TO 7:PRINT UP$;:NEXT XX
3090 GOTO 2000
3500 PRINT HOM$;
3510 PRINT "Program exited..."
3520 PRINT "Control relinquished to basic monitor."
3530 END
```



Panel "Key-Push" Clock Generation Circuit

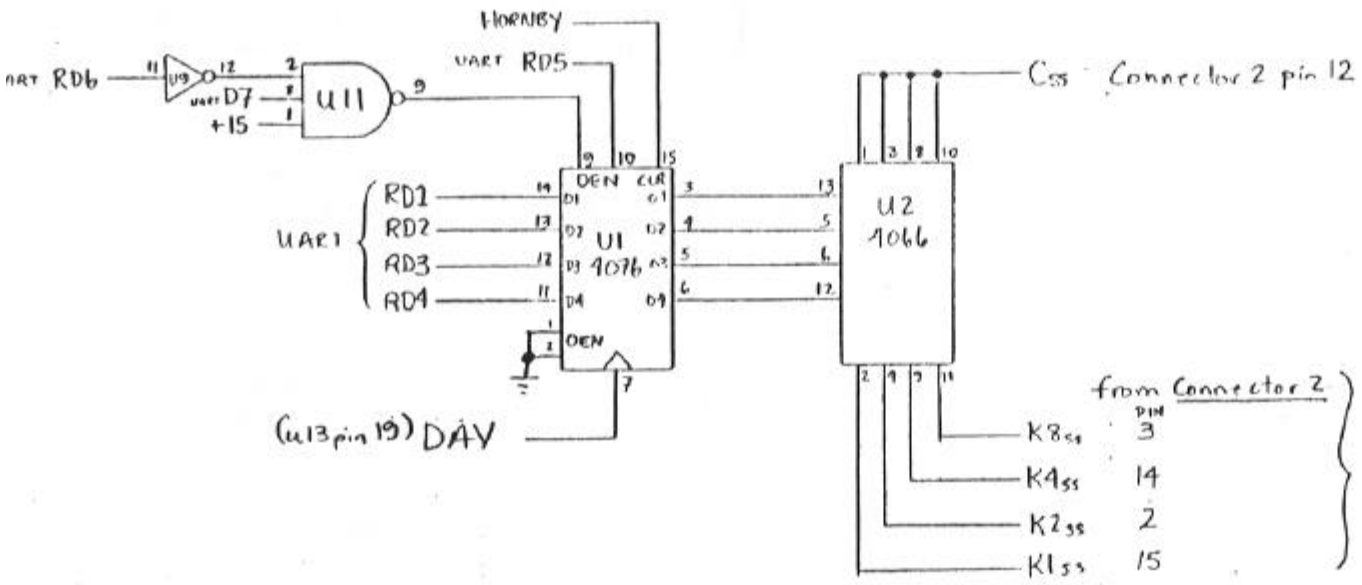
WG# 001 Rev B	Clock Ckt. Diagram
RS-232 Interface to HORNBY ZERO-1	

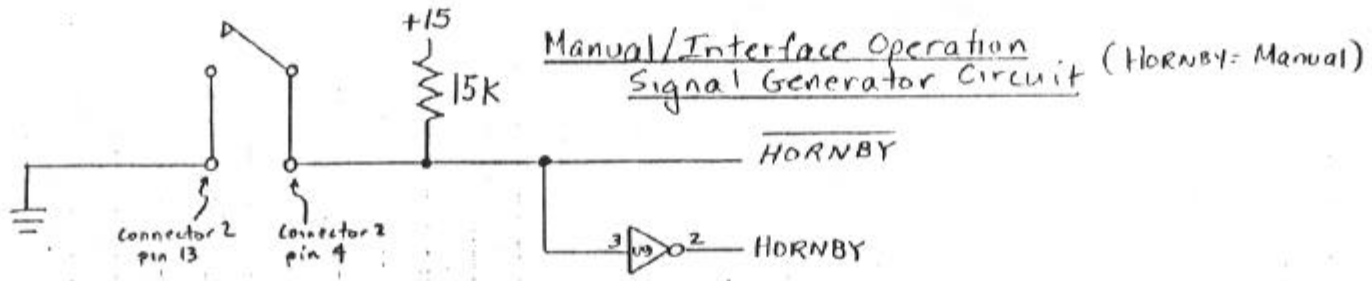


DWG #002 RevB | Keyboard Control Ckt. Diagram

RS-232 Interface to HORNBY Zero-1
Model Train Controller

Speed Control Circuit

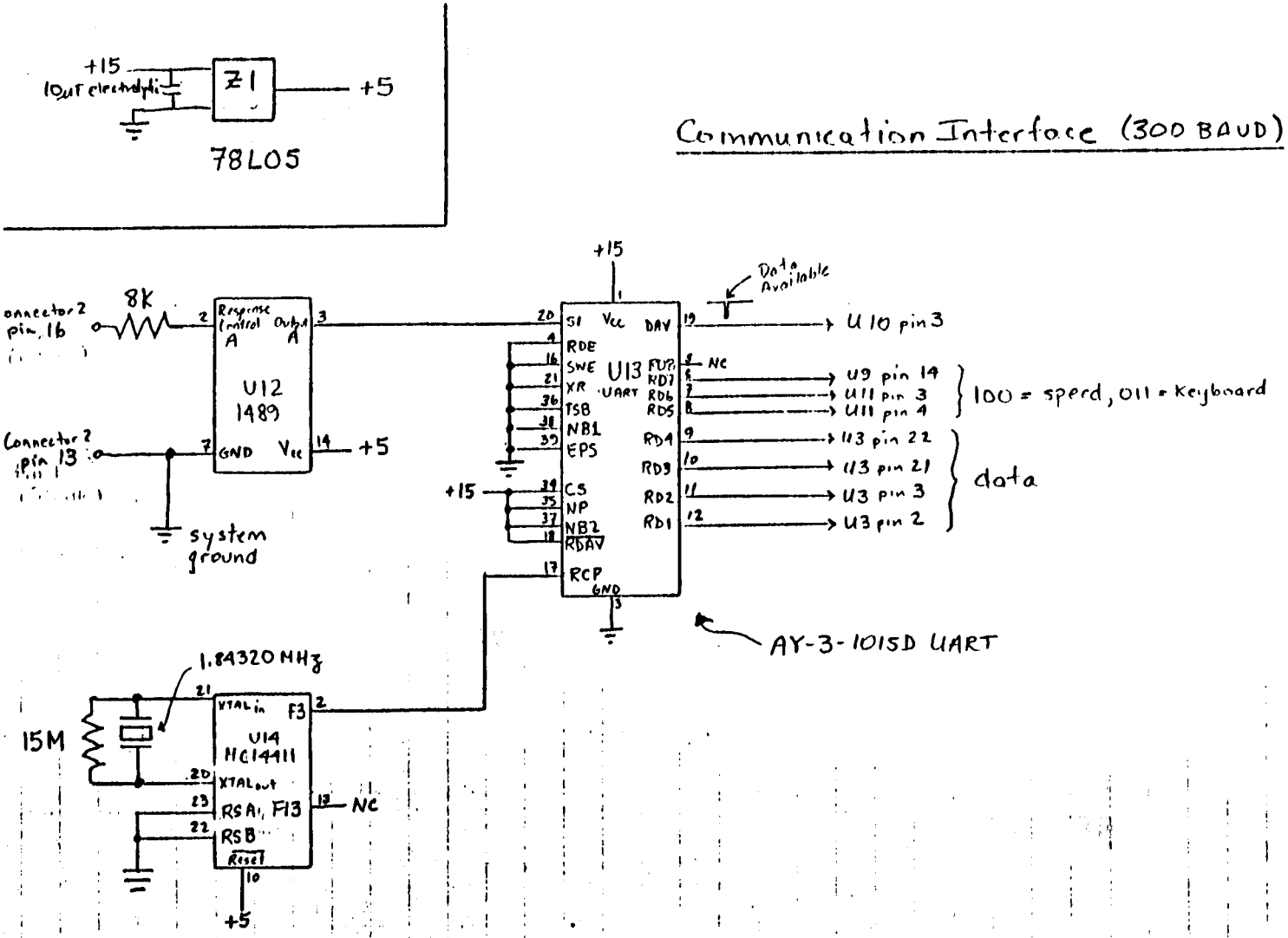




Manual/Interface Operation Signal Generator Circuit

(HORNBY = Manual)

NG # 003 Rev. B	Circuit Diagram
RS-232 Interface to HORNBY ZERO-1	

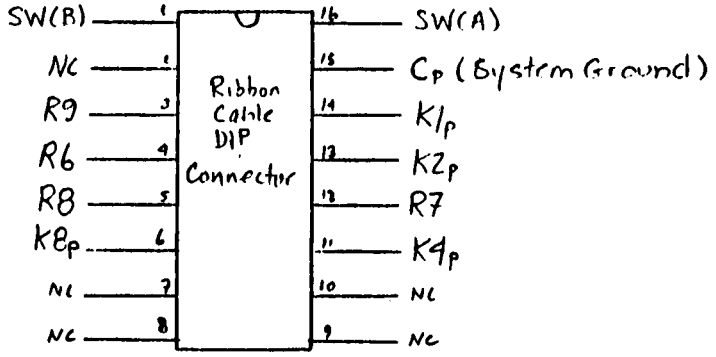


Note: For 110 BAUD, RSB = +15, RSA = GND, use F13 (pin 13) for the output pin.

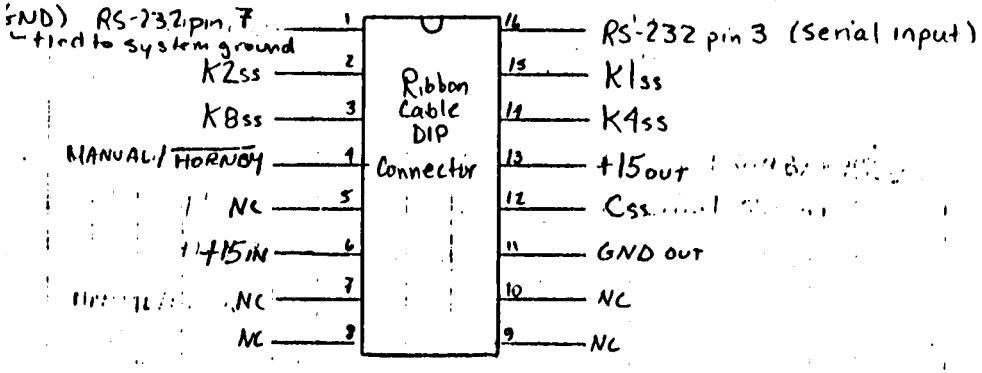
DWG #004 Rev. A | Circuit Diagram

RS-232 Interface to HORNBY ZERO-1 Model Train Controller

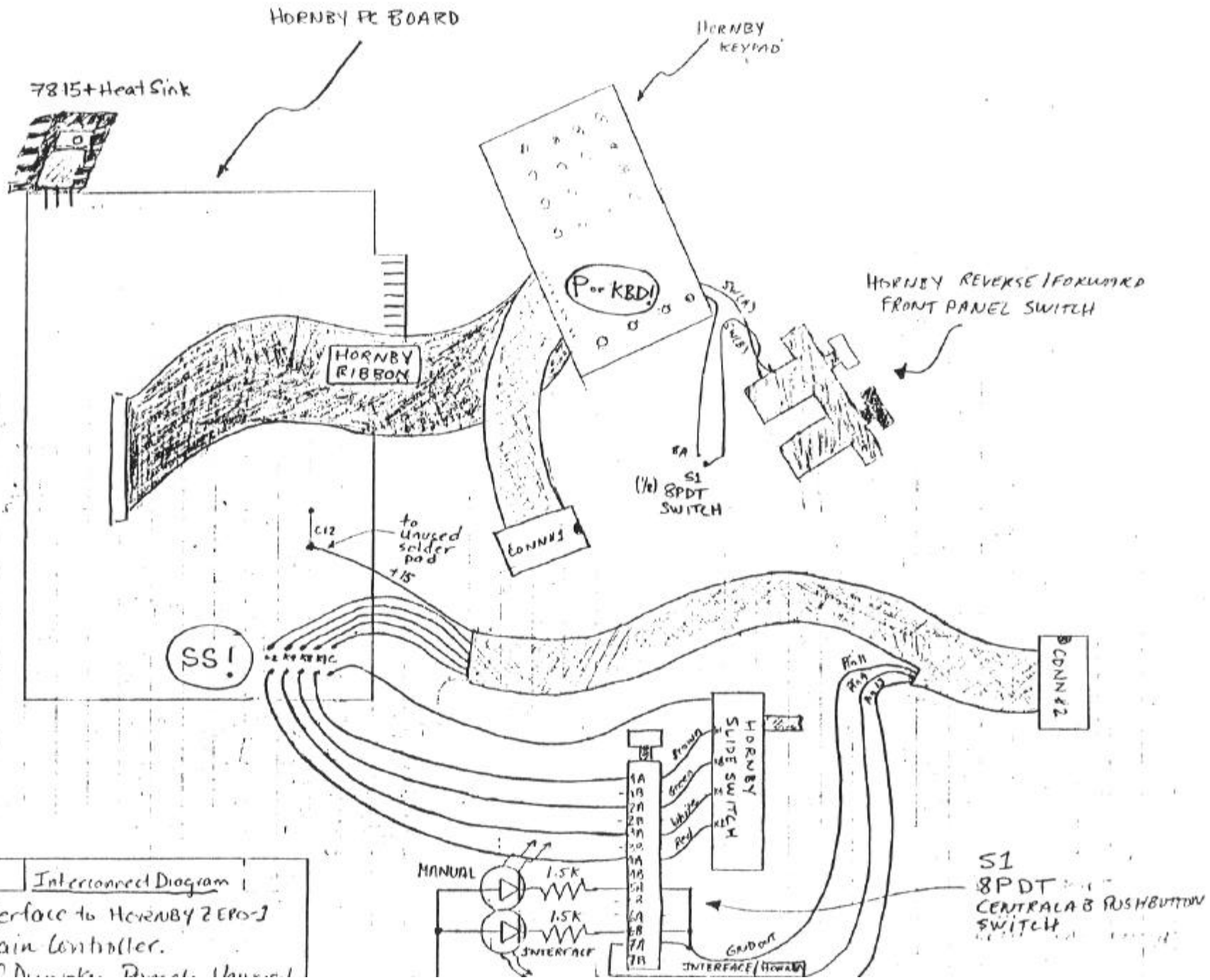
Connecter 1 - to Keyboard



Connector 2 - to Slide Switch solder points and +15V all on HORNBY Pc Board, RS-232 input, and external switch connections.



WG 1005 Rev A.	Connector Diagrams
RS-232 Interface to HORNBY ZERO-1	



NG #006 Rev. A	Interconnect Diagram

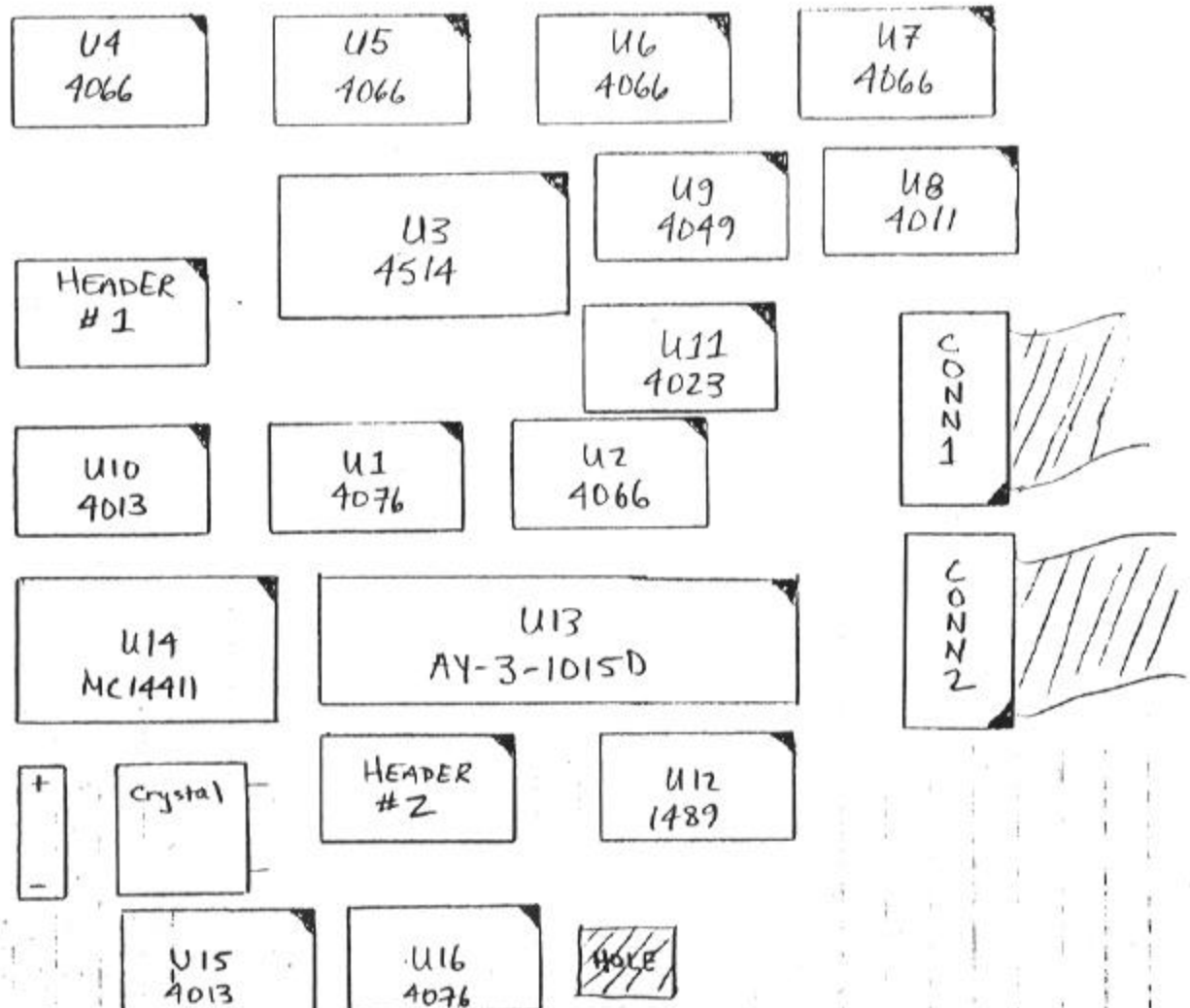
RS-232 Interface to HORNBY ZERO-1 Model Train Controller.

QTY.	PART	CIRCUIT ID NUMBER(S)	POWER AND GROUND CONNECTIONS: +15	GND	DESCRIPTION
2	4076	U1, U16	pin 16	pin 8	CMOS QUAD LATCH
5	4066	U2, U4, U5, U6, U7	14	7	CMOS QUAD BILATERAL SWITCH
1	4514	U3	24	12	CMOS 1 of 16 MUX, OUTPUTS NORMALLY LOW
2	4011	U8, U15	14	7	CMOS QUAD 2-INPUT NAND
1	4049	U9	1	8	CMOS HEX INVERTER
1	4013	U10	14	7	CMOS DUAL DFF
1	4023	U11	14	7	CMOS TRIPLE 3-INPUT NAND
1	1489	U12	14 (+5)	7	QUAD MDTL LINE RECEIVER
1	AY-3-1015D	U13	1	3	UART
1	MC14411	U14	24 (+5)	12	BIT RATE GENERATOR
1	78L05	Z1	NA	NA	+5v LOW POWER VOLTAGE REGULATOR
1	7815	Z2	NA	NA	+15V VOLTAGE REGULATOR
1	81F242 (PBS SERIES)	S1	N/A	NA	8PDT ALTERNATING ACTION CENTRALAB SWITCH

Notes: (1) +15, gnd, except where marked otherwise.

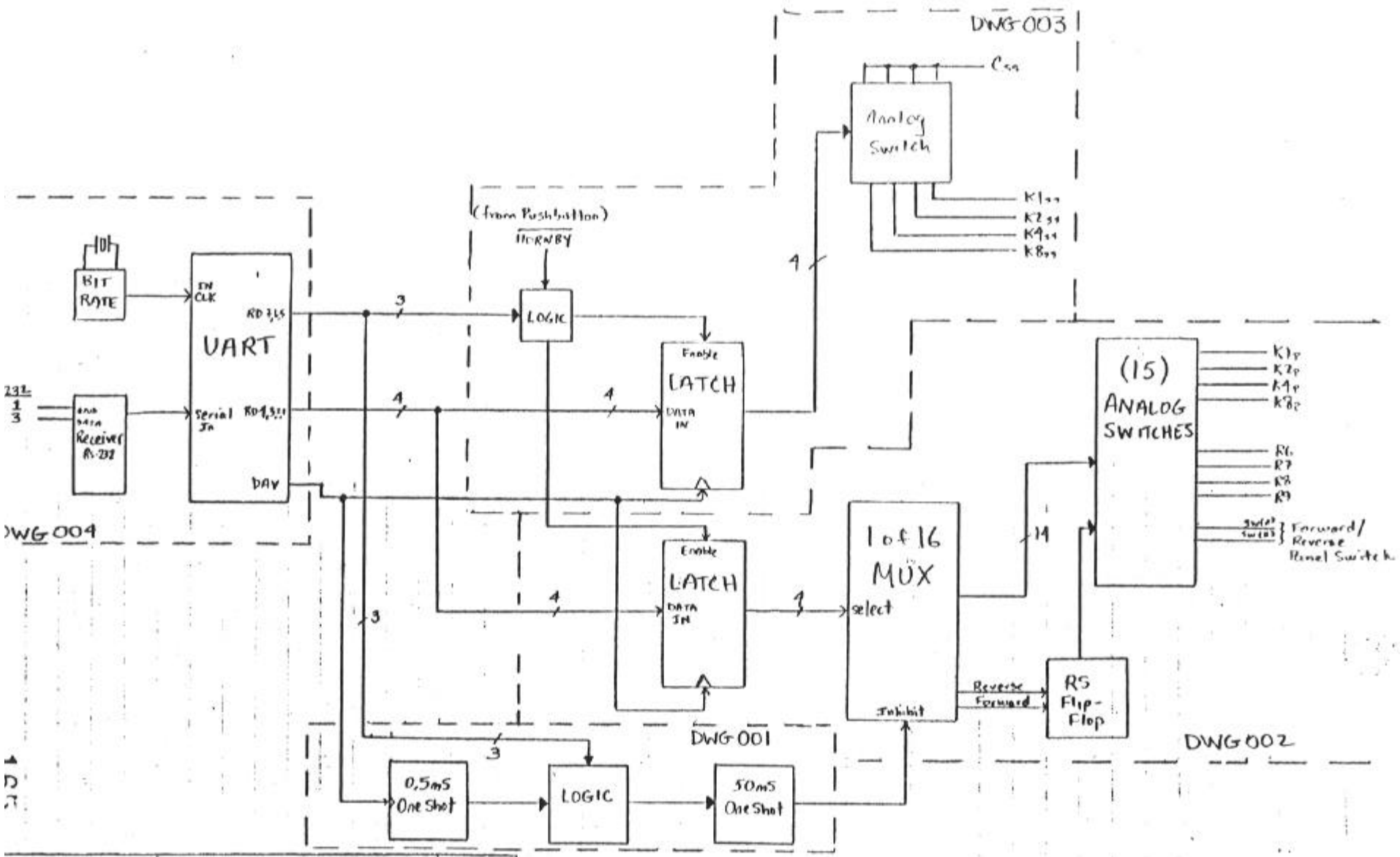
G# 008 RevA | Parts List

RS-232 Interface to HORNBY ZERO-1



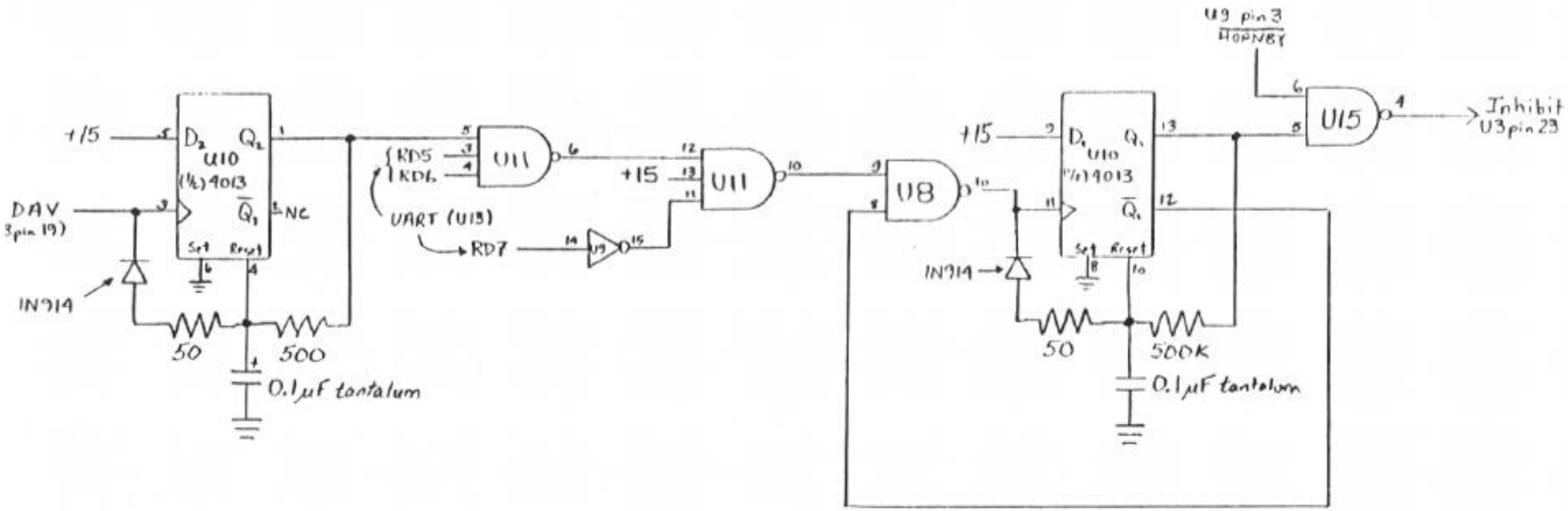
UG #009 Rev A	Component Layout

RS-232 Interface to HORNBY Zero-1
Model Train Controller



G # 007 Rev A | Block Diagram

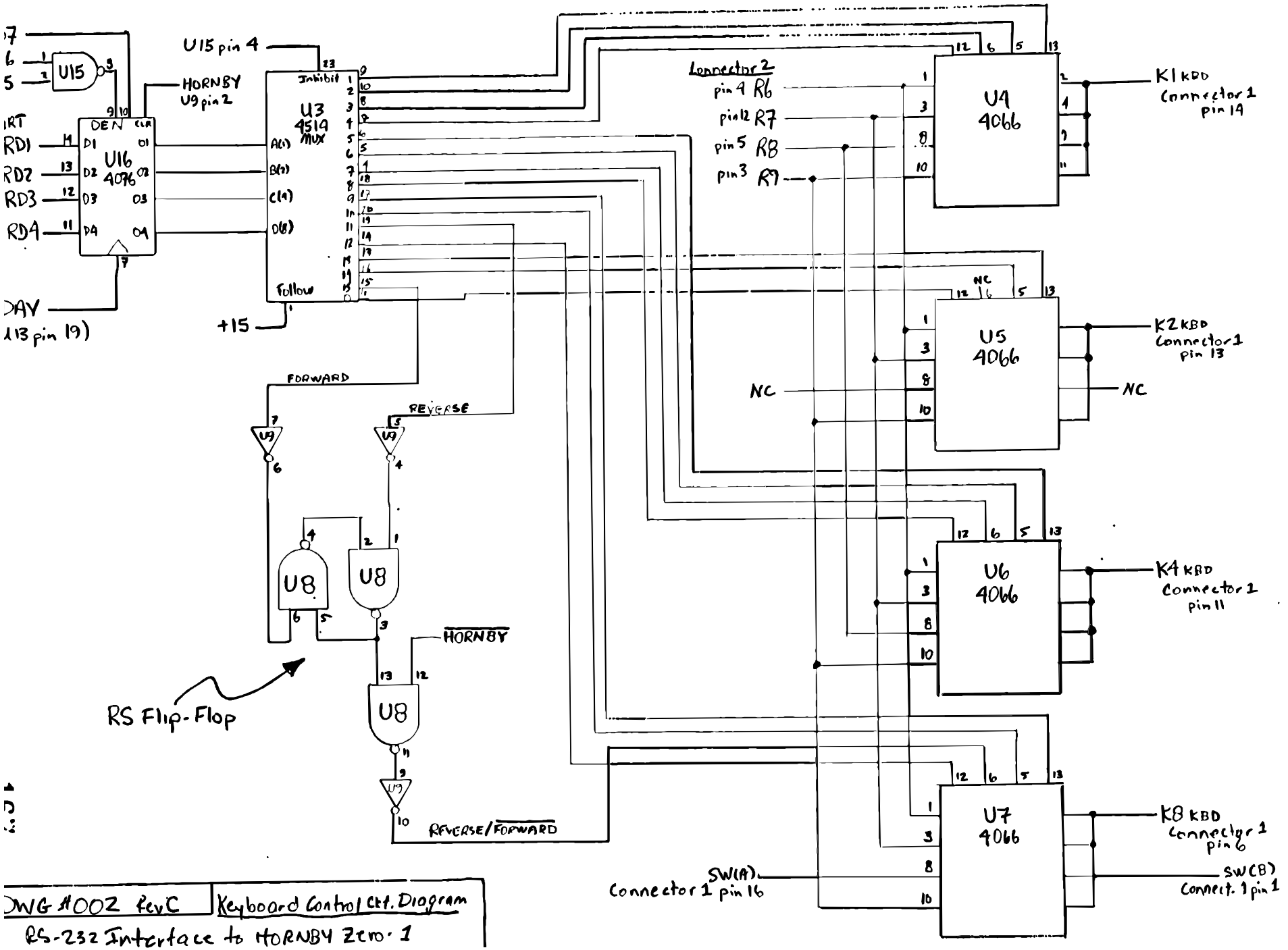
RS-232 Interface to HORNBY ZERO-1 Model Train Controller



Panel "Key-Push" Clock Generation Circuit

NG # 001 Rev C	Clock CKT Diagram

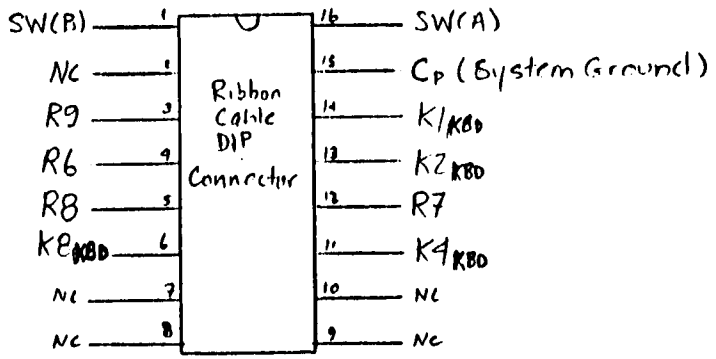
RS-232 Interface to HORNBY ZEPO-1
Model Train Controller



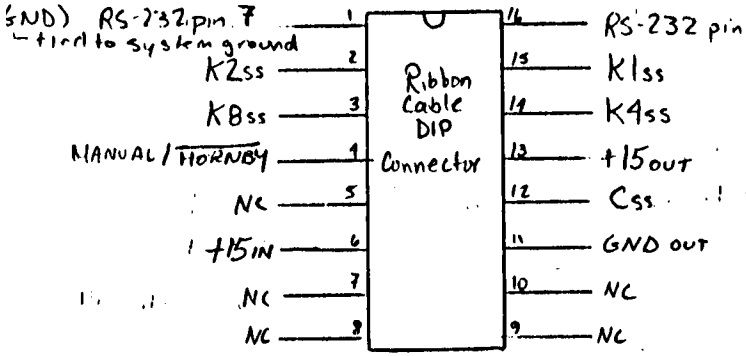
DWG #002 RevC | Keyboard Control Ckt. Diagram

RS-232 Interface to HORNBY Zero-1

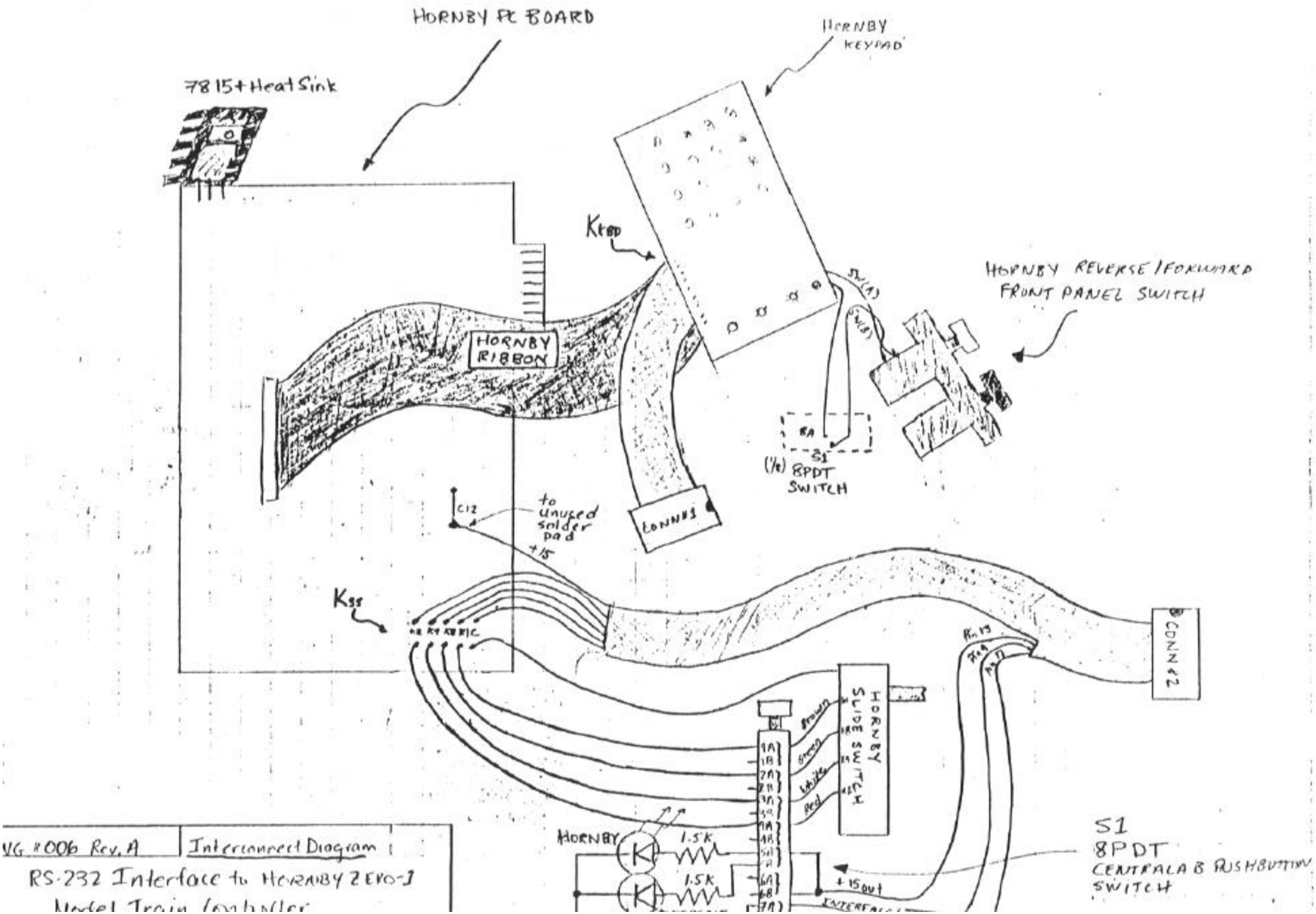
Connector 1 - to Keyboard



Connector 2 - to Slide Switch solder points and +15V all on HORNBY Pc Board, RS-232 input, and external switch connections.

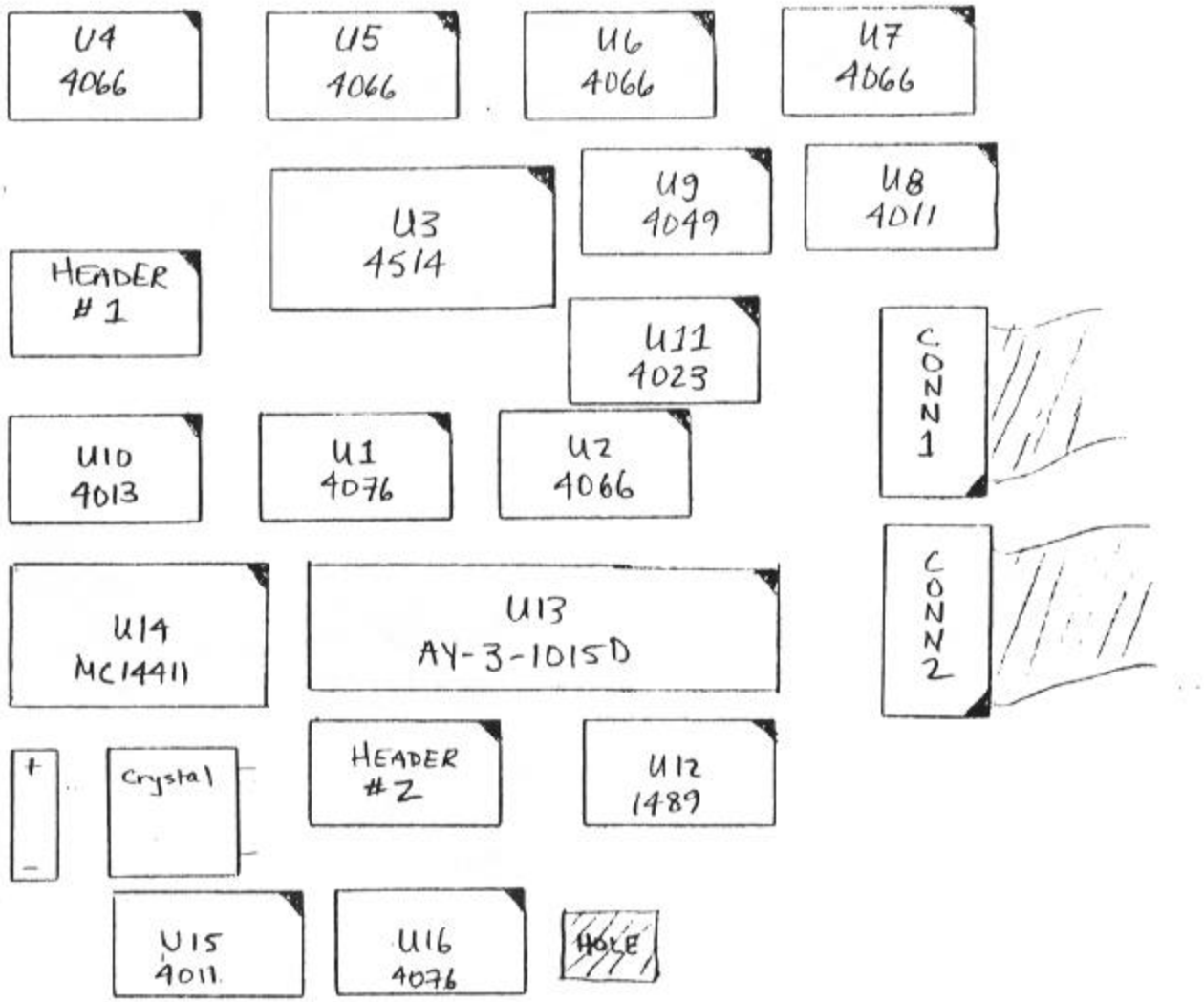


ANG #0015 Rev A	Connector Diagrams
RS-232 Interface to HORNBY ZERO-1 Model Train Controller	



VG #006 Rev. A	Interconnect Diagram

RS-232 Interface to HORNBY ZERO-1
Model Train Controller



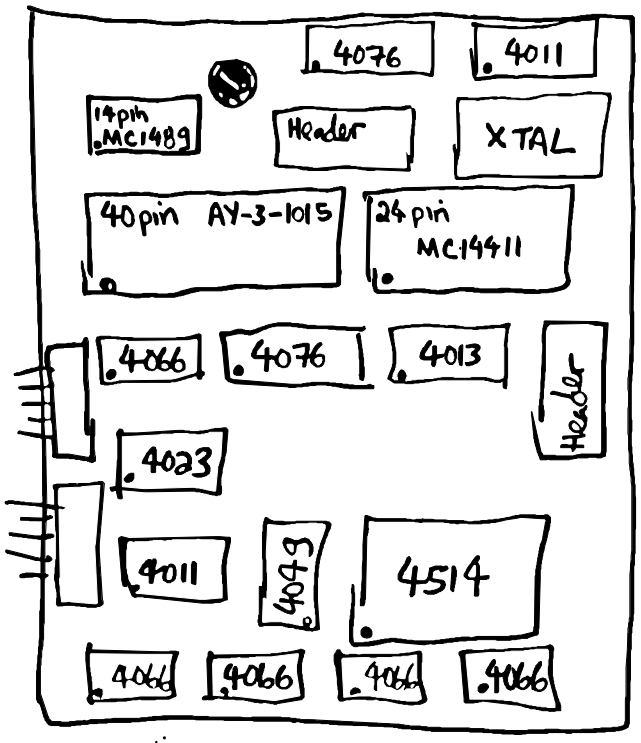
NG #009 Rev B | Component Layout

RS-232 Interface to HORNBY Zero-1
Model Train Controller

KEY	TMS1000 OUTPUT	TMS1000 INPUT
PANIC	R5	$K1_{KBD}$
1	R6	$K1_{KBD}$
2	R7	$K1_{KBD}$
3	R8	$K1_{KBD}$
4	R9	$K1_{KBD}$
5	R6	$K4_{KBD}$
6	R7	$K4_{KBD}$
7	R8	$K4_{KBD}$
8	R9	$K4_{KBD}$
9	R6	$K8_{KBD}$
INERTIA	R7	$K8_{KBD}$
SLAVE	R8	$K8_{KBD}$
LOCO	R9	$K8_{KBD}$
←	R6	$K2_{KBD}$
→	R7	$K2_{KBD}$
C	R8	$K2_{KBD}$
Ø	R9	$K2_{KBD}$

component side — — circuit prototype sent to Hornby / TrainTronics for evaluation

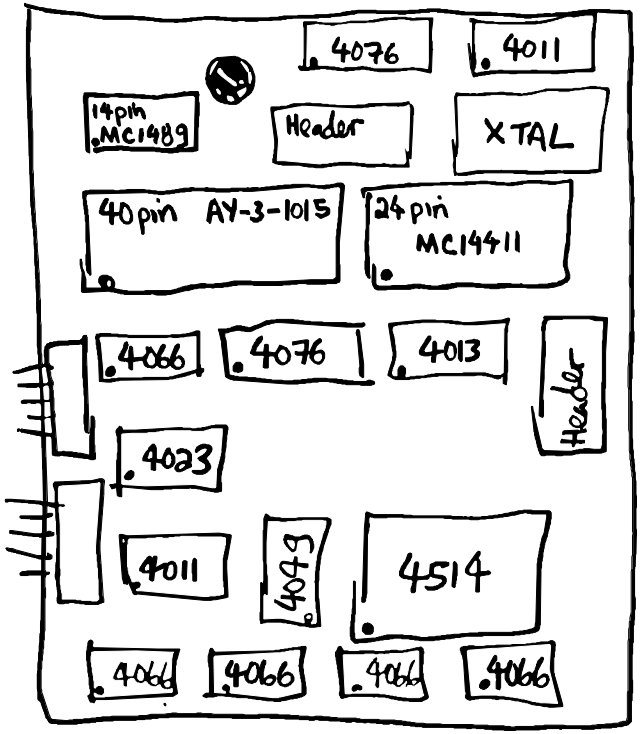
20 Jan 83



MGL

component side — — circuit prototype sent to Hornby / TrainTronics for evaluation

20 Jan 83



MGL