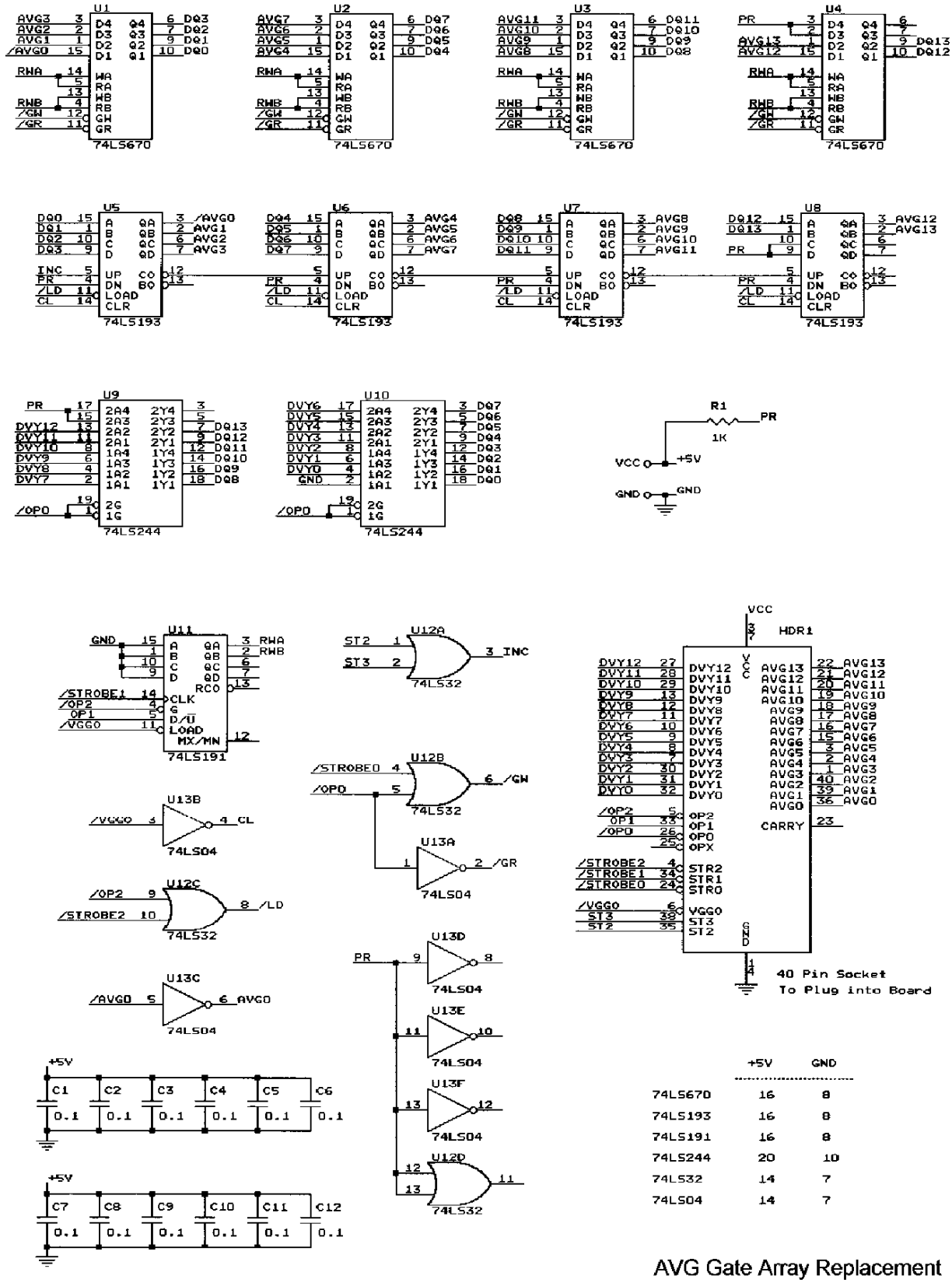
Figure 24 - Vector Generator Gate Array



	+5V	GND
74LS670	16	8
74LS193	16	8
74LS191	16	8
74LS244	20	10
74LS532	14	7
74LS04	14	7

AVG Gate Array Replacement
Jed Margolin 3/18/2001

VECTOR GENERATOR STACK AND PC PROCESSOR

1.0 SCOPE

1.1 General - This procurement specification defines the requirements for an integrated circuit gate array using Complementary MOS (CMOS) process technology.

1.2 Functional Description The gate array is designed to be the Stack/Stack Jointer and Program Counter for the analog vector generator circuit used in vector type games. The Program Counter (PC) has 14 Bits with Carry Out. The Stack is three levels deep and stores the upper thirteen bits (AVGI - AVGI3) of the Program Counter. (Refer to Figure 2). When the gate array receives control signals from the analog vector generator it executes the JMP, JSR, and RTS Instructions. See attached detail timing diagrams for proper operation. The normal gate array operation is an Increment PC (INC PC) operation which is controlled by ST3 and ST2 signals supplied by external hardware. For the JMP Instruction, /STROBE_2 loads the PC with address data on DVYO- DVY12. For the JSR Instruction, /STROBE_O stores the current PC data onto the Stack, /STROBE_1 Increments the Stack Pointer and /STROBE_2 Loads the PC with DVYO - DVY12 data. For the RTS Instruction, /STROBE_1 Decrements the Stack-Pointer and /STROBE_2 Loads the PC from Stack Outputs. ST2 (LOW) Enables normal Increment PC operation. On the next rising edge of ST3, AVGO will go back to HIGH. The rising edge of AVG0 will increment PC's AVG1 - AVGI3 outputs.

The Stack Pointer is designed to count in the 0, 1, 2, 0, 1, 2 ... etc. sequence, however, since power on can cause the Stack Pointer value to be 3, it is recommended that a "dummy" JSR Instruction be generated before using a real JSR Instruction.

The vector generator circuit has nine instructions with each instruction consisting of either 2 bytes or 4 bytes of data. The op-code field of each instruction resides in the second byte. Since the op-code must be fetched before any instruction parameter field, the Program Counter was specifically designed with AVG0-AVG13 following a 1, 0, 3, 2, 5, 4 ... etc. count sequence. During an instruction op-code fetch cycle, AVG0 is HIGH. After an instruction op-code is fetched, ST2 goes LOW which causes AVG0 to go LOW after the next ST3 clock. This allows the vector generator to fetch the Instruction parameters for current instruction. As long as AVG0 is LOW, the next ST3 clock will bring AVG0 back to HIGH. This LOW to HIGH AVG0 transition will increment AVGI - AVGI3 and consequently the Program Counter will be ready for the next instruction fetch.