Fig. 12 SINGLE TONE WITH ENVELOPE SHAPE/CYCLE PATTERN 1010 (R15=12_8, all other registers same as Fig. 10)

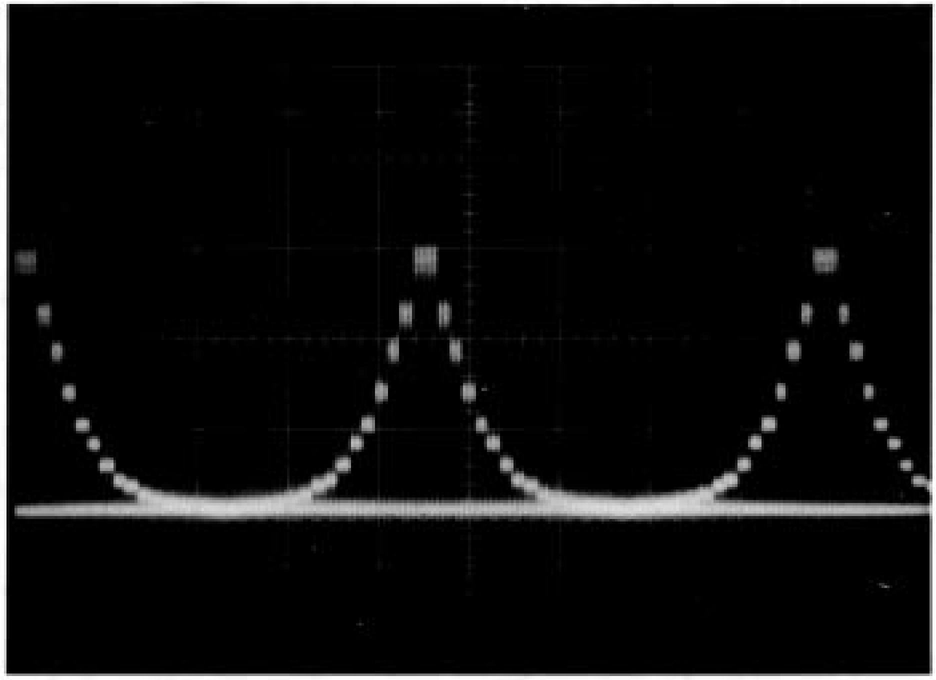
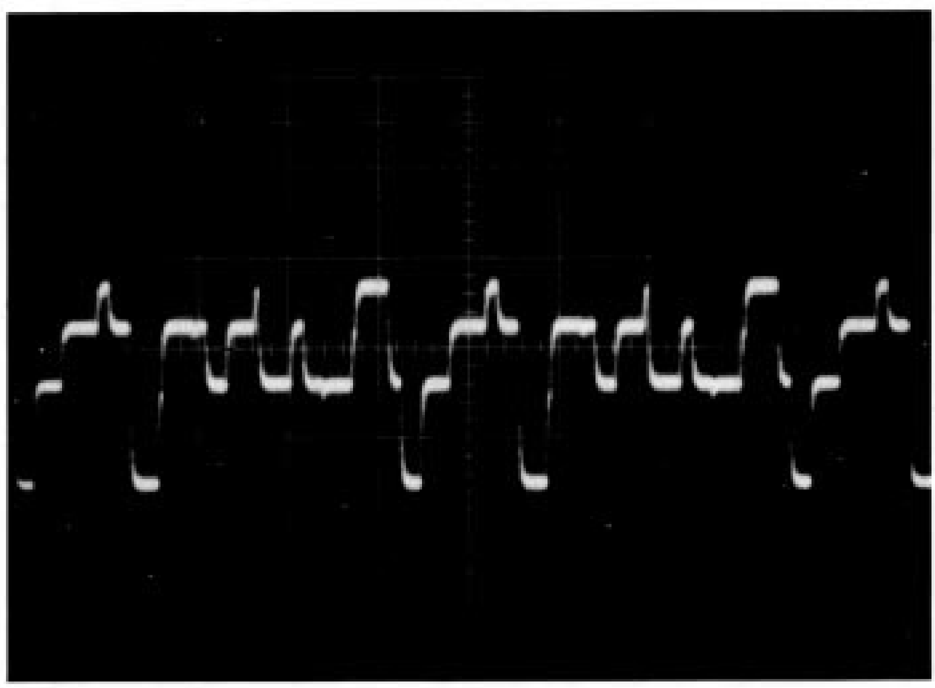


Fig. 13 MIXTURE OF THREE TONES WITH FIXED AMPLITUDES



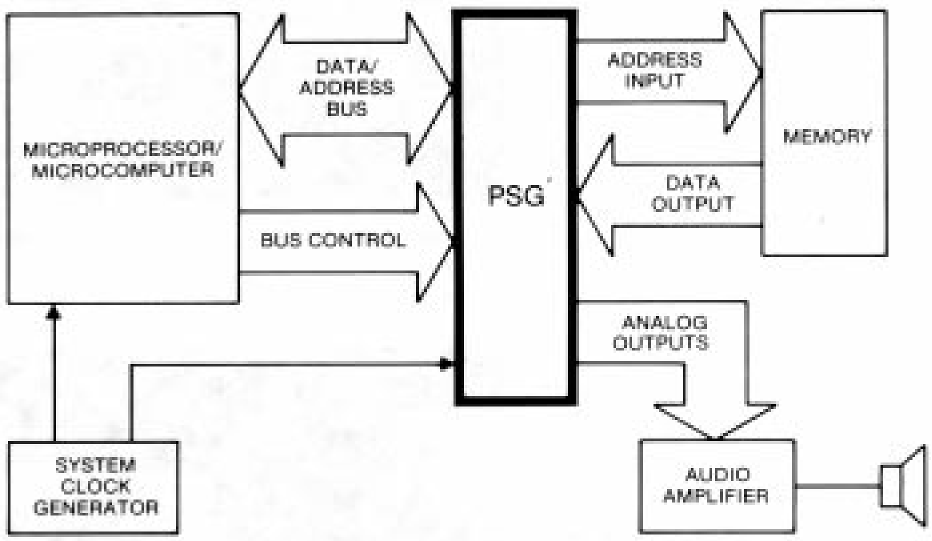
4 INTERFACING

4.1 Introduction

Since the AY-3-8910/8912 PSG must be used with support components, interfacing to the circuit is an obvious requirement. The PSG is designed to be controlled by a microprocessor or microcomputer, and drive directly into analog audio circuitry. It provides the link between the computer and a speaker to provide sounds or sound effects derived from digital inputs.

The following paragraphs provide examples and illustrations showing the ease with which an AY-3-8910/8912 Programmable Sound Generator may be utilized in a microprocessor/microcomputer system.

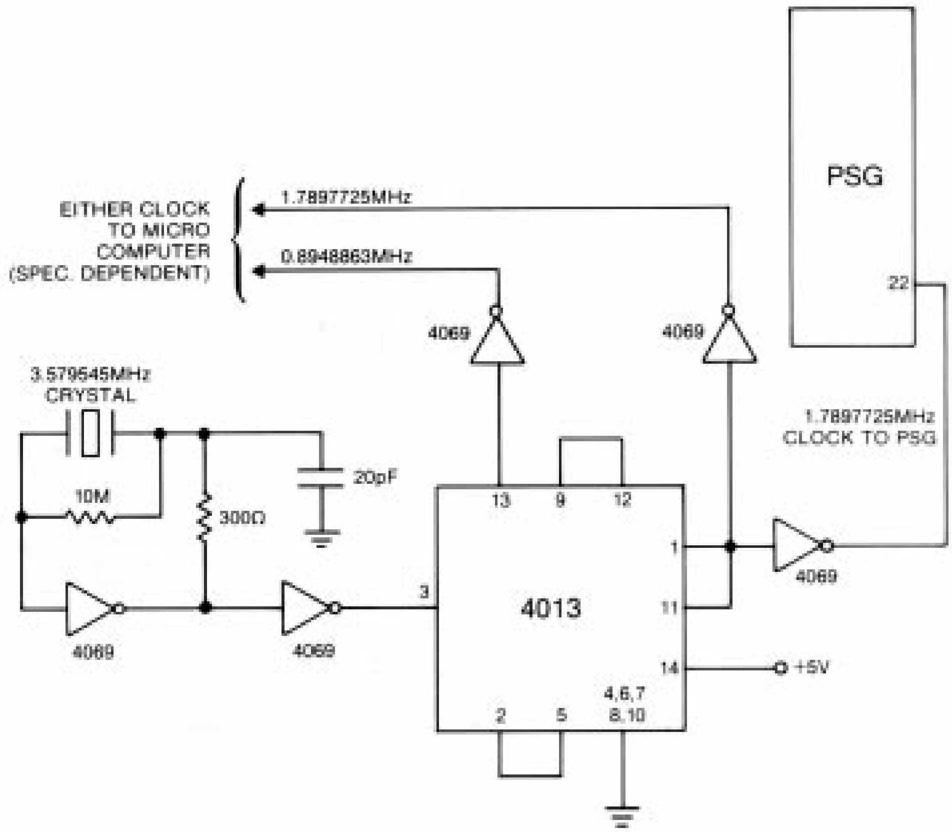
Fig. 14 SYSTEM BLOCK DIAGRAM



4.2 Clock Generation

An economical solution to providing a system clock is shown in Fig. 15. It consists of a 3.579545MHz standard color burst crystal, a CD4069 CMOS inverter, and a CD4013 to divide the color burst frequency in half. The clock produced for the PSG runs at a 1.7897725MHz rate. Depending on the microcomputer used, its clock should be selected within its specified value.

Fig. 15 CLOCK GENERATION

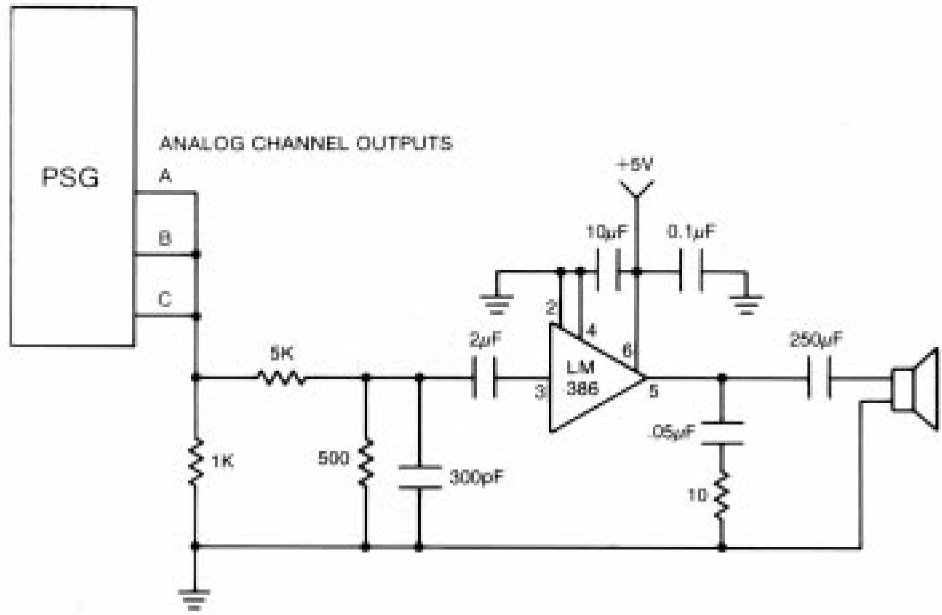


4.3 Audio Output Interface

Fig. 16 illustrates the audio output connections to a commercially available LM386 audio amplifier. It shows channels A, B, and C summed together to enable complex waveforms to be composed and amplified through a single external amplifier. These channels may be individually amplified through separate channels for more exotic sound systems.

Each output channel is individually controlled by separate amplitude registers (R10, R11, R12) and an enable register (R7) in the PSG.

Fig. 16 **AUDIO OUTPUT INTERFACE**



4.4 External Memory Access

The ROM or PROM shown connected to the PSG in Fig. 17 illustrates an option for providing additional data information for processor support. The two I/O registers within the PSG are used in this case to address the memory via I/O Port A (8 Bits) and read data from the memory via I/O Port B (8 Bits).

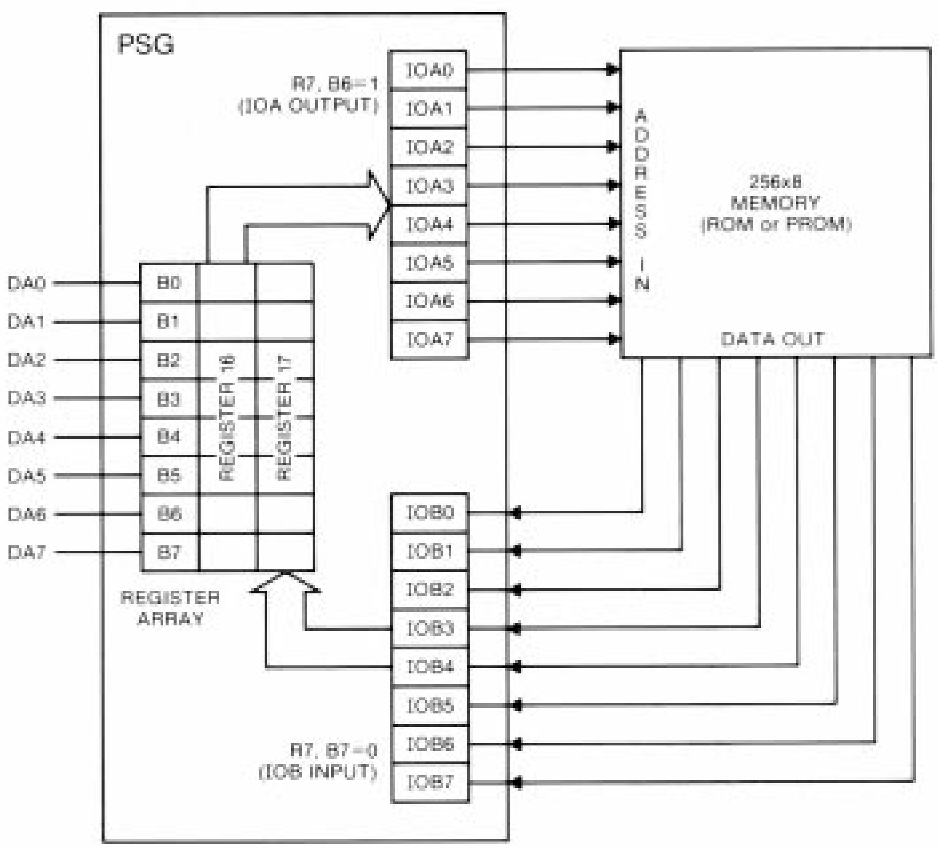
An example of the bus control sequence to address and read an external memory connected to I/O ports A and B would be as follows (Assume Port A addresses and Port B reads):

	Bus Codes			
Bus Control	**BDIR**	**BC2**	**BC1**	**Explanation of Bus Data (DA7--DA0)**
Latch address	1	1	1	00000111: Latch R7 to program I/O Ports
Write to PSG	1	1	0	01000000: Set B7, B6 to 0, 1 respectively
Latch address	1	1	1	00001110: Latch R16 to address memory
Write to PSG	1	1	0	00000001: Address data to memory
Latch address	1	1	1	00001111: Latch R17 to read memory
Read from PSG	0	1	1	XXXXXXXX: Memory data contained in R17

NOTE: BC2 in the above Bus Codes may be permanently tied to +5V thus requiring only two bus control lines for all control operations (refer to Section 2.3 for a complete explanation).

Also, RAM or EAROM may be used in place of the ROM or PROM shown by altering the program to use PORT B as an I/O. Port B then will be able to write data as an output and read data as an input.

Fig. 17 EXTERNAL MEMORY ACCESS



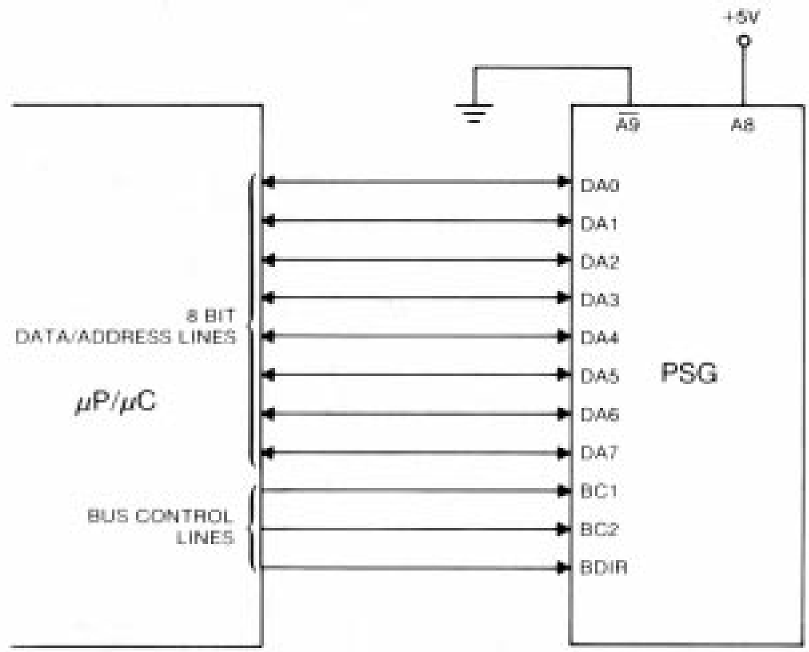
4.5 Microprocessor/ Microcomputer Interface

In Fig. 18, the lines identified DA7--DA0 are the input/output bus bits 7--0. This 8 bit bus is used to pass all data and address information between the AY-3-8910/8912 and the system processor.

BC1, BC2 and BDIR are bus control signals generated by the processor to direct all bus operations. These operations are identified as Latch Address, Write to PSG, Read from PSG, and Inactive.

The following Sections detail specific interfaces to several popular microprocessors/microcomputers.

Fig. 18 MICROPROCESSOR/MICROCOMPUTER INTERFACE



4.6 Interfacing to the PIC 1650

Fig. 19 shows the schematic of an AY-3-8910 demonstrator circuit. This configuration uses a PIC 1650 as the main controller in the circuit. The PIC 1650 is used to scan the keyboard, fetch data from the PROMs, write data to the AY-3-8910 and provide the timing for the AY-3-8910.

The interfacing is direct since the PIC 1650 and the AY-3-8910 operate with compatible supplies and input/output voltages.

This particular schematic illustrates how a microcomputer with additional memory can produce a stand-alone music and sound effects circuit. The circuit as shown operates with manual keyboard selections.

As Fig. 19 shows, the design for the interface connects directly to the output pins of the 1650 and the BC1, BC2, BDIR pins. The software then has the responsibility of manipulating these signals to signal the PSG to perform the proper address latch, read or write operations.

The program routine in this section illustrates code which is used in a hand-held demonstrator unit. This demonstration unit illustrates the range of PSG capabilities, including music, sound effects and I/O control. Note that the generalized routines perform the address latching before every read for convenience.

The "READ ROM" routine illustrates use of the generalized read and write routines to access the outside world through the PSG to read and write.

4.6.1 WRITE DATA ROUTINE

```
                ;WRITE FROM 1650 TO 8910
                ;ADDRESS OF 8910 REG IN 'ADDRES'
                ;DATA TO WRITE IN 'DATA'
024  0066  WRIT1  MOVWF  ADDRES    ;
025  1026  WRITE  MOVF   ADDRES,W  ;GET REGISTER NO.
026  0045         MOVWF  IOA       ;SET ADDRESS
027  1006         MOVF   IOB,W     ;GET PRESENT BC1, BC2, BDIR ETC.
030  7370         ANDLW  370
031  6404         IORLW  4         ;SET BAR
032  0046         MOVWF  IOB       ;SEND BAR
033  7370         ANDLW  370
034  0046         MOVWF  IOB       ;SEND NACT
035  1027         MOVF   DATA,W
036  0045         MOVWF  IOA       ;PUT DATA ON D/A PINS OF 8910
037  1006         MOVF   IOB,W
040  7370         ANDLW  370
041  6406         IORLW  6
042  0046         MOVWF  IOB       ;SEND DWS
043  7370         ANDLW  370       ;SET UP NACT
044  0046         MOVWF  IOB       ;SEND NACT
045  4000         RET              ;RETURN TO CALLING ROUTINE
```

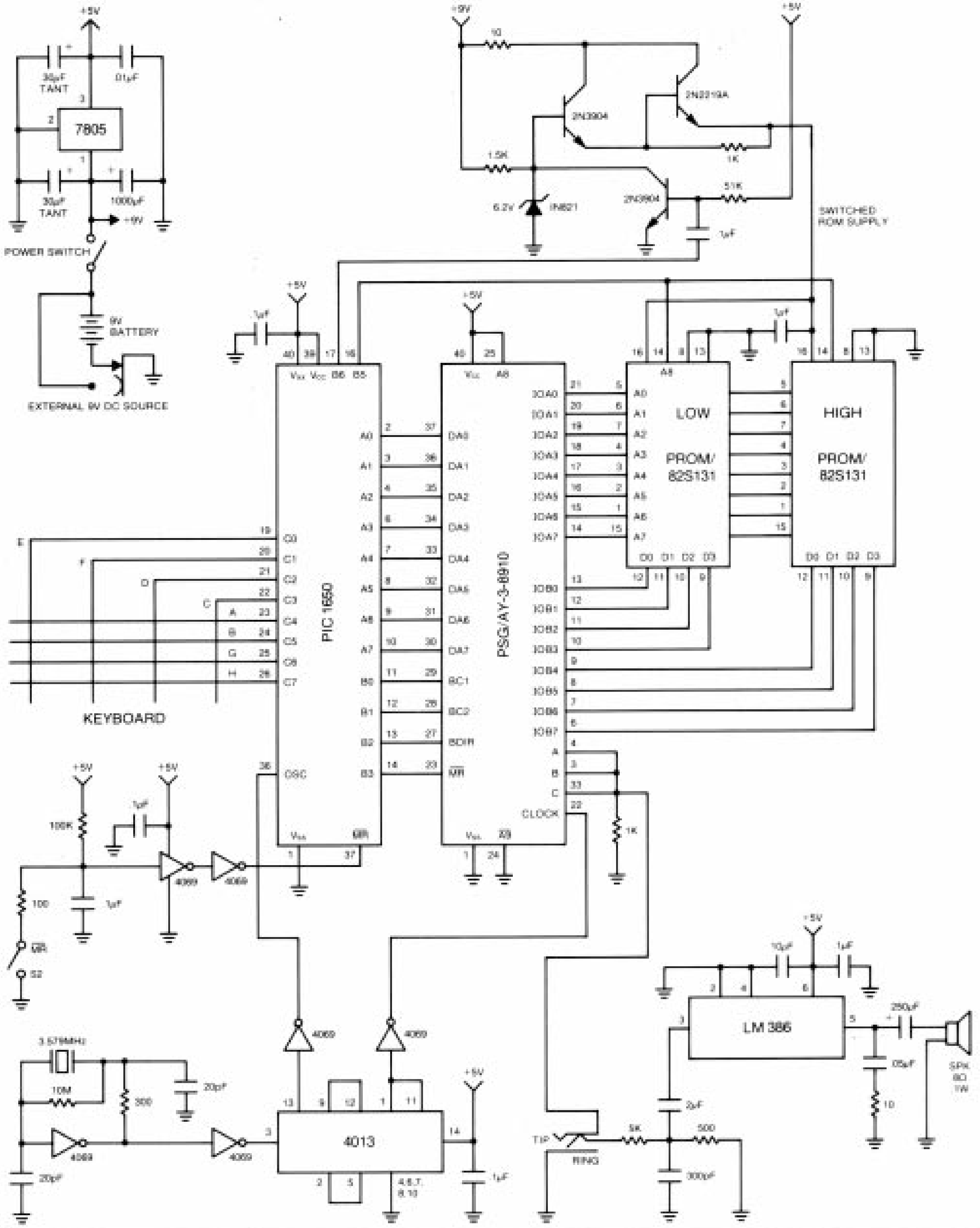
4.6.2 READ DATA ROUTINE

```
                 ;ADDRESS OF READ IN REGISTER 'ADDRES'
                 ;AFTER READ, INPUT DATA IN REGISTER 'DATA'
                 ;ENTRANCE READ1 ASSUMES THAT REGISTER NUM IN W
                 ;
000  0066  READ1 MOVWF  ADDRES    ;BYPASS ADDDRESS STORE
001  1026  READ  MOVF   ADDRES,W  ;GET REGISTER NO.
002  0045        MOVWF  IOA       ;MOVE TO 8910 D/A PINS
003  1006        MOVF   IOB,W     ;GET PRESENT BC1,BC2,BDIR ETC.
004  6404        IORLW  4         ;SET BAR
005  0046        MOVWF  IOB       ;SEND BAR
006  7370        ANDLW  370
007  0046        MOVWF  IOB       ;SEND NACT
010  6377        MOVLW  377
011  0045        MOVWF  IOA       ;SET FOR INPUT
012  1006        MOVF   IOB,W
013  7370        ANDLW  370
014  6403        IORLW  3         ;SET DTB
015  0046        MOVWF  IOB       ;SEND DTB
016  1005        MOVF   IOA,W
017  0067        MOVWF  DATA      ;SAVE DATA
020  1006        MOVF   IOB,W
021  7370        ANDLW  370
022  0046        MOVWF  IOB       ;SEND NACT
023  4000        RET              ;RETURN TO CALLING ROUTINE
```

4.6.3 READ ROM ROUTINE

```
                  ;ADDRESS OF ROM IN W AT ENTRANCE NEXROM
                  ;ADDRESS OF ROM IN ROMAD AT ENTRANCE ROMRD
                  ;
                  ;INCREMENTS ROMAD AFTER READ. IF ROM ADDRESS
                  ;CROSSES 256 BORDER, MAKE UPPER BANK SELECT=1
                  ;
                  ;USES 8910 REG 16 FOR ADDRESS
                  ;8910 REG 17 FOR INPUT DATA
046  1030  NEXROM MOVF   ROMAD,W
047  0067  ROMRD  MOVWF  DATA     ;PUT ADDRESS
050  6016         MOVLW  16       ;I/O A ADDRESS
051  0066         MOVWF  ADDRES
052  2306         BCF    IOB,6    ;TURN ON ROM
053  4425         CALL   WRITE    ;SEND TO IOA
054  1266         INCF   ADDRES   ;TO IOB ADDRESS
055  4401         CALL   READ     ;GET DATA
056  2706         BSF    IOB,6    ;TURN OFF ROM
057  1770         INCFSZ ROMAD    ;TO NEXT LOC
060  4000         RET
061  2646         BSF    IOB,5    ;SET HIGH SELECT
062  4000         RET             ;RETURN TO CALLING ROUTINE
```

Fig. 19 PIC 1650/AY-3-8910 SYSTEM EXAMPLE



4.7 Interfacing to the CP1600/1610

As shown in Fig. 20, the wiring is direct between the AY-3-8910 and a CP1600/1610 microprocessor. The levels are compatible thus eliminating any need for level converters. Even the terminology between the IC's remains constant to provide simple-to-follow connections.

The CP1600/1610 acts as a controller in this configuration fetching data from ROM's contained elsewhere in the system. The CP1600/1610 also acts as the bus controller developing the necessary timing for the AY-3-8910.

4.7.1 WRITE DATA ROUTINE

The program necessary to write to a selected register is as follows:

```
        MVI value, R0; move in value to be written
        MVO R0, Reg; write to register
```

The routine to load all registers with the same value is as follows:

```
        MVII Reg 0, R4
        CLRR R0
Here MVO@ R0, R4
        CMPI Reg 0 + 17, R4
        BLT Here
```

4.7.2 READ DATA ROUTINE

The routine to read from a selected register is as follows:

```
        MVI Reg, R0; get data from reg in R0
        MVO R0, value; store in memory
```