ChipCorder®
TECHNOLOGY BY ISD

APPLICATIONS NOTE 1

Using the ISD33000 Device with a Microcontroller

This application note describes how to use the ISD33000 device with a microcontroller, illustrating the ease of use with any SPI compatible microcontroller.

Unlike other families of single-chip record and playback from ISD, the ISD33000 is a microcontroller slave peripheral device. It is controlled with a microcontroller either with an SPI compatible hardware already on the microcontroller or with I/O ports that emulate SPI protocol. For the ease of use and illustration a microcontroller with an SPI compatible serial interface is used in this application note.

SERIAL PERIPHERAL INTERFACE (SPI)

ISD33000 operates from an SPI serial interface. The SPI protocol is a synchronous serial data transfer protocol. The SPI interface has 4 control I/O pins.

1. Slave Select ($\overline{SS}$): This pin when low will select the ISD33000.
2. Master Out Slave In (MOSI): This is the serial input to the ISD33000.
3. Master In Slave Out (MISO): This is the serial input to the ISD33000.
4. Serial Clock (SCLK): This is the clock input to the ISD33000. The master microcontroller provides this clock.

The data transfer protocol assumes that the microcontroller's SPI shift registers are clocked on falling edge of the SCLK. On the ISD33000, data is clocked in on the MOSI input on the positive clock edge and data is clocked out on the MISO output on the negative clock edge.

All serial data transfers begin with the falling edge of $\overline{SS}$ pin. Slave Select is held low during all serial interface and it is held high between instructions. Each operation that ends in an EOM (End of Message) or an OVF (Overflow) from the ISD33000 will generate an Interrupt. The interrupt will be cleared the next time an SPI cycle is initiated. As the interrupt data is shifted out of the ISD33000 MISO pin, control and address data are simultaneously shifted into the ISD33000 MOSI pin. Care should be taken such that the data shifted in is compatible with current system operation. An operation begins with the RUN bit set and ends with the RUN bit reset. All operations begin with the rising edge of the $\overline{SS}$.

SPI CONTROL REGISTER

The SPI control register provides control of individual device functions such as play, record, stop/pause, message cueing (fast forward), power-up, power-down, start and stop operations and ignore address pointers.

There are five control bits associated with the ISD33000 that control the device. These bits are:

C0 = MC

When this bit is set to 1 during playback it starts a message cueing cycle (Fast forward to the next EOM). C0 operation is not defined during Record.

C1 = IAB

When this bit is set to 1, any address data shifted into the SPI MOSI shift register is ignored. A resulting Record, Playback or Message Cueing operation

will begin at the next address in the device memory at the end of the proceeding operation.

When IAB (Ignore Address Bit) is set LOW, a playback or record operation starts from address (A9–A0) and ends at the end of that row. If no other control data is input to the SPI port, the device will continue that operation over again at the same address and therefore "loop" on that row. To continue playback or record consecutively through the memory, a second SPI cycle should immediately be input with the IAB bit changed HIGH before the device reaches the end of a row. To stop an operation it is important to note that if the IAB bit is LOW, the data shifted into address bit locations will be transferred to the device's internal address register and the device's internal address prior to the stop instruction will be lost.

It is recommended to set IAB = HIGH when a stop/pause command is issued if address read back is desired.

C2 = PU

When this bit is set HIGH, the device powers-up and is ready for an operation after T_{PUD} (approximately 25 ms). Wait T_{PUD} before issuing an operational command. It is important to wait T_{PUD} after the power-up command is issued before sending another command; the device will not function properly if a record or playback command with power-up command is issued in the same SPI cycle. For example to record from address 00 the following program cycle should be used.

1. Send power-up command (00100<X-X)
2. Wait T_{PUD} (per device specification, approximately 25 ms)
3. Start recording (10100<A9–A0>)
4. Continue consecutive recording (10110 <X-X>)
5. C3 = P/R. When this bit is set HIGH, the device goes to playback mode and when it is LOW the device goes to record mode.
6. C4 = RUN. When this bit is pulled HIGH, the device begins an operation and when LOW, stops an operation.

SPI OPCODE FORMAT

The ISD33000 accepts either an 8-bit command or 16-bit command from a microcontroller through the SPI port. The opcode format is 5 control bits and 11 address bits for a 16-bit command. The opcode format is 5 control bits and 3 address bits for an 8 bit command with the 3 address bits being "don't care" bits.

EXAMPLES OF RECORD/PLAYBACK OPERATION

RECORD AT AN ADDRESS

Power Up the Device

Send 00100<xxxx> to the SPI

Wait T_{PUD}

Send 00100<xxxx> to the SPI

Wait 2 x T_{PUD}

Start Recording at <10 bit address >

Send 10100<x><10bit address> (note; IAB = 0)
Send 10110<xxxx> (note; IAB = 1)

Recording will continue until memory is full or a new SPI cycle with RUN = 0 is input or until the end of memory is reached (Overflow Interrupt).

Stop Record and Power down

Send 00000<xxxx> to the SPI

OR

Stop/Pause Record and Do Not Power-Down

Send 0 0110 <xxxx>

RECORD A MESSAGE AT THE NEXT AVAILABLE ADDRESS

If the Device is Not Powered Up

Send 00100<xxxx> to the SPI

Wait T_{PUD}

Send 00100<xxxx> to the SPI

Wait 2 x T_{PUD}

Record the Next Message

Send 10110<xxx>

Recording continues until ready to stop (or device runs out of memory space)

PLAYBACK AT AN ADDRESS

Power Up the Device

Send 01100<xxx> to the device.

Wait T_{PUD}.

Start Playback at <10 bit Address>

Send 11100<x><10 bit address>.

Send 11110<xxx> to continue playback.

Playback continues until memory overflows or ready to end (reaches EOM).

Stop playback and power down.

Send 01010<xxx>.

OR

Stop Playback don't power down.

Send 01110 <xxx>.

PLAYBACK AT THE NEXT MESSAGE

If the device is Not Powered Up

Send 01100<xxxx>

Wait T_{PUD}

Playback the next message

Send 11110<xxx>

Playback continues until an EOM is reached (or chip runs out of memory space)

PLAYBACK THE "3RD" MESSAGE (FAST FORWARD TO MESSAGE NUMBER 3 AND PLAY IT)

Power up the device

Send 01100<xxx>

Wait T_{PUD}

Start Message Cueing Cycle at "0" Address

Send 11101<x><00 0000 0000>

Send 11111<xxx>

Device runs at 800 times normal play speed, audio muted.

Device stops at next EOM, gives EOM interrupt, increments address counter 1 count, now the address pointers points to '2nd" Message. Execute second message cueing cycle at "next" message.

Send 11111<xxx>

Device runs at 800 times normal play speed, audio muted.

Device stops at next EOM, gives EOM interrupt, incumbents address counter 1 count, now the address pointers points to '3rd" message. Play "next" message.

Send 11110<xxx>

The third message will play at normal speed. When message ends EOM interrupt occurs

READ INTERRUPT STATUS BITS AND CURRENT ADDRESS

Clock in Instruction

Send <xxxxx><xxx>

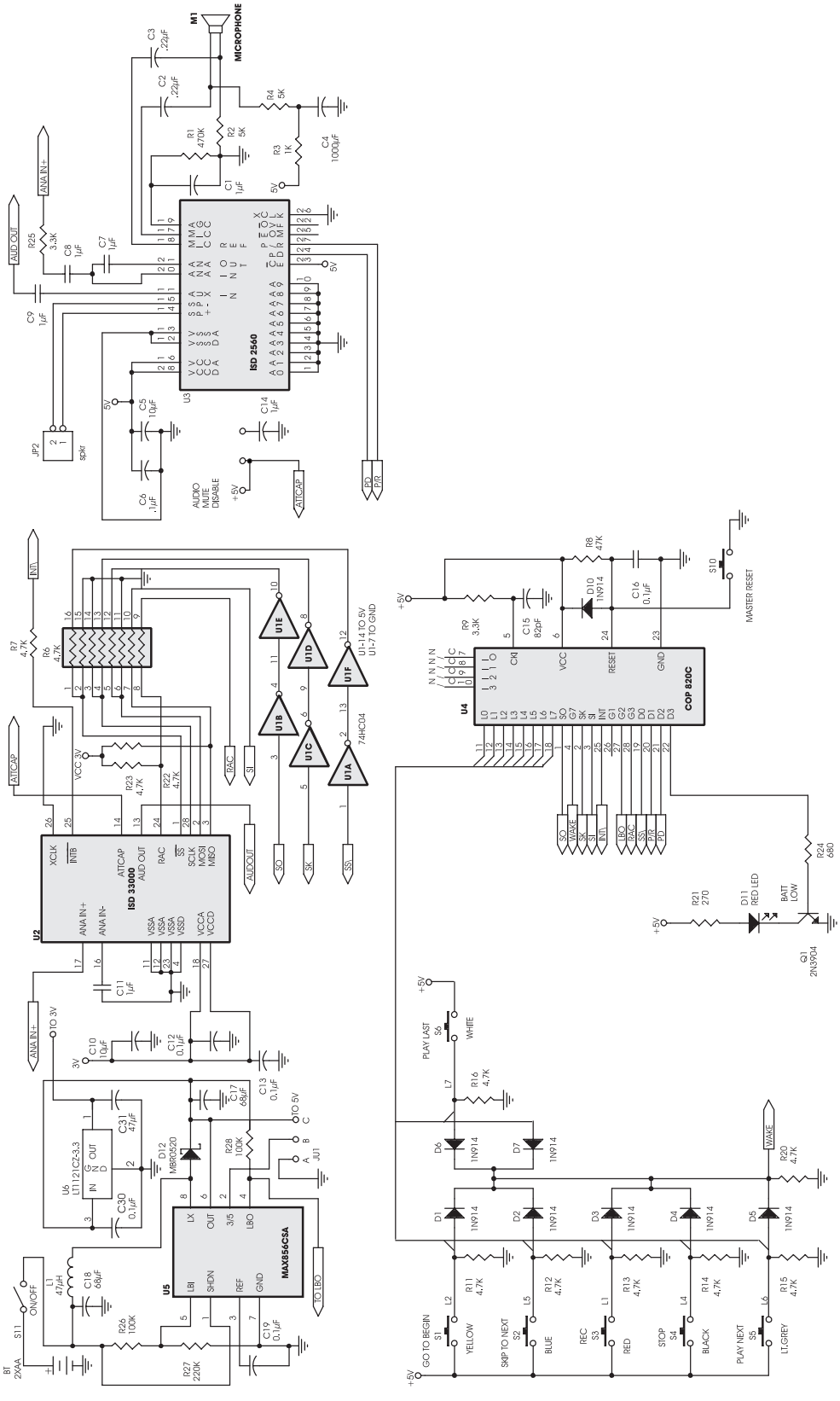
Device will execute the operation as specified by the instruction.

If there is no desire to change the status of the device, care should be taken that the command is compatible with current operation.

Send SPI 8 clocks to read the status bits (OVF and EOM)

Send SPI 16 clocks to read status and current address.

Figure 1: ES302 Schematic



NOTE: *L1 is a Sumida CD43-470. C17 and C18 are 10V Tantalum (SMD). D12 substitute 1N817 SCHOTTKY.*

APPLICATION EXAMPLE (ISD33000 DEMO BOARD)

The ISD-ES302 Demo Board is designed to enable listeners hear the voice quality of the 3-volt part and to demonstrate some of the features available through the SPI control bus.

FEATURES OF THE ISD-ES302 DEMO BOARD

- Record
- Play—last
- Play—next
- Stop/pause
- Go-to- beginning of memory
- Fast forward through messages.

HARDWARE

The three major integrated circuits on the board are the ISD33000 voice chip, U2, the NSC COP820C microcontroller, U4, and an audio chip, U3. The audio chip is an ISD2560 of which only the microphone preamplifier and the speaker driver are used. It is run on five volts whereas the ISD33000 is run on three volts. The audio chip would not normally be needed because the audio input and speaker output functions are already on the board in a cellular phone, for example. The board is powered by two AA cells. These cells drive a voltage converter to get the five volts for the audio chip and the microcontroller. This voltage is then regulated down to three volts (or 3.3 VDC) for the ISD33000 device.

SOFTWARE

COP8 is an 8-bit microcontroller from National Semi Conductor with 1 Kbytes of on-chip ROM and 64 bytes of on-chip RAM. The serial interface on-chip "microwire" is compatible with the ISD33000 SPI port.

The COP microcontroller provides the interface between the user's fingers on the keyboard and the ISD33000 device. It interprets the key strokes and issues the appropriate commands to the ISD33000. It also monitors the ISD33000 status registers and interrupt output to keep track of operations. The program also handles any interrupt that is generated by the ISD33000 during playback.

The following flowchart is used to create the demo software. The interrupt is a background routine. Every time an EOM or OVF is detected the ISD33000 will generate an interrupt. The microcontroller issues a stop command when EOM is detected and when an OVF is detected, the microcontroller re-initializes the address pointer to beginning of the memory in play mode followed by a stop command.

Figure 2: Demo Software Flowchart

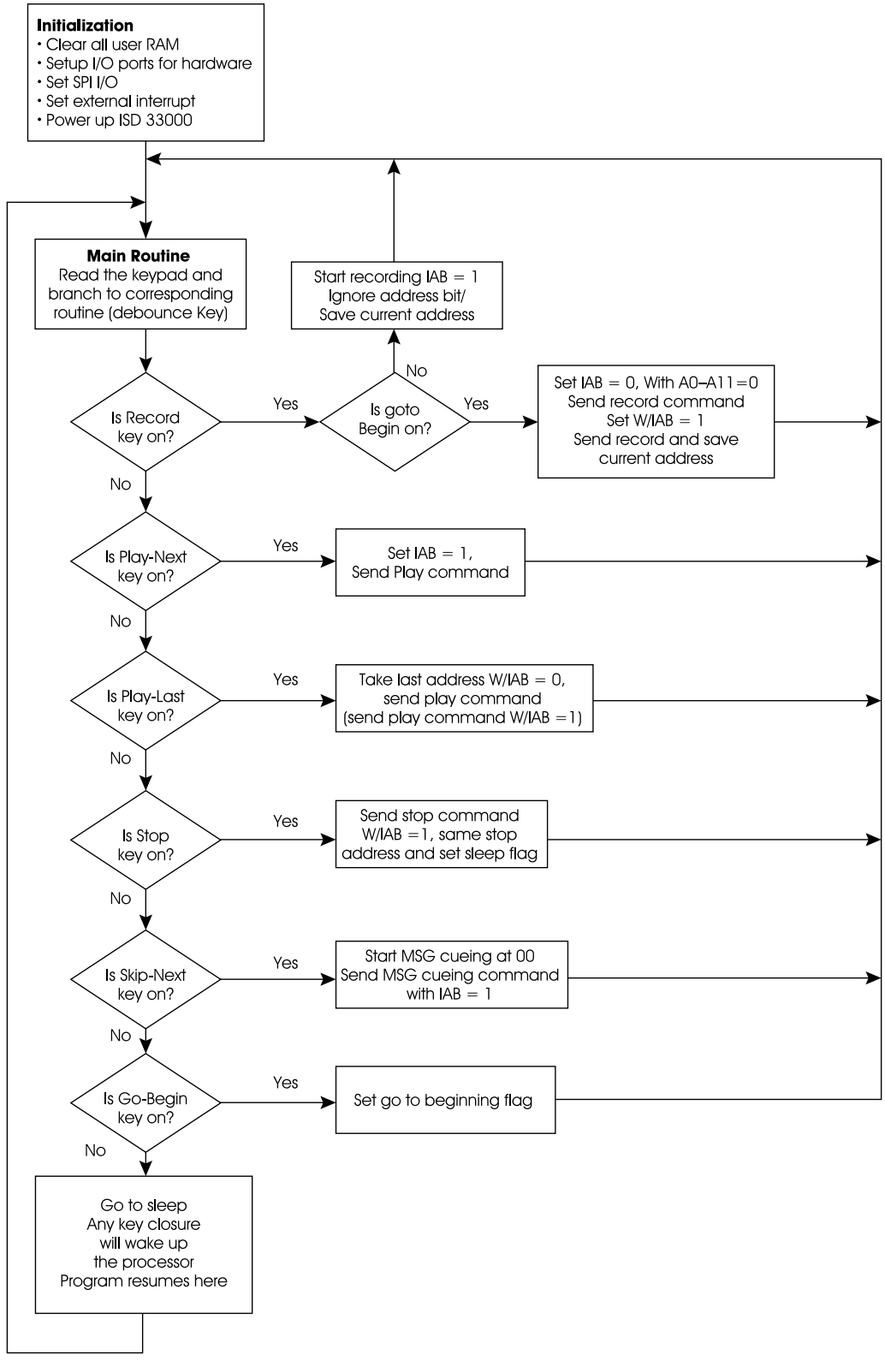
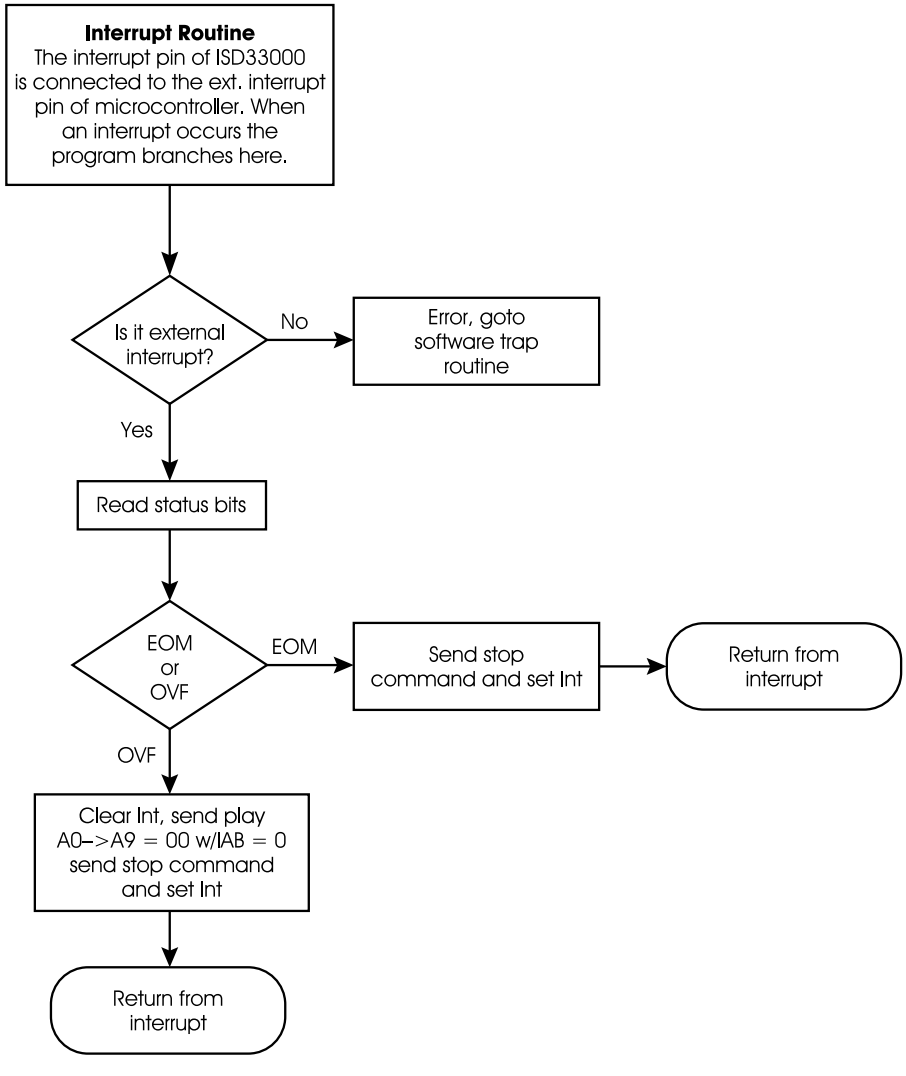


Figure 3: Interrupt Routine Flowchart



```
;************************************************************************
;*   ISD33000  D E M O      P R O G R A M                               *
;*    S E C O N D   V E R S I O N                                        *
;*                                                                       *
;*    (C) 1996 I N F O R M A T I O N   S T O R A G E   D E V I C E S    *
;*                                                                       *
;*                                                                       *
;*      MICROCONTROLLER      : COP820C                                   *
;*      DATE OF LAST REVISION: 06.03.96                                  *
;*      NAME OF SOURCE FILE  : demo.asm                                  *
;*      CLOCK SPEED          : 2.8 - 2.2 MHz                            *
;;************************************************************************
;************************************************************************
.list 33
.INCLD COP820.INC     ;SYMBOLS USED FREQUENTLY BY COP820
.FORM
;Assignment For ISD33000 DEMO
;
;* CONSTANT DECLARATIONS
    SS      =       0       ; SLAVE SELECT
    D2      =       2       ; D PORT PIN 2
    D1      =       1       ; D PORT PIN 1
    LED     =       3       ; LED INDICTOR PIN
    LBO     =       2       ; LOW BAT DETECTOR
;************************************************************************
;REGISTERS
;
    DELC2   =       0F0     ; COUNTER FOR SOFTWARE TIME LOOPS
    DELC1   =       0F1     ; COUNTER FOR SOFTWARE TIME LOOPS
    DBU     =       0F2     ; COUNTER FOR DEBOUNCE TIME
;   0FC                     ; RESERVED FOR X POINTER
;   0FD                     ; RESERVED FOR STACK POINTER
;   0FE                     ; RESERVED FOR B POINTER
;MEMORY
;************************************************************************
    STAT    =       0C      ; STATUS REGISTER
    NEW     =       00      ; STOP OR COMPLETION
    OVF     =       01      ; OVERFLOW
    BEG     =       02      ; BEGGIN OF MEMORY
    HALT    =       03      ; OPERATION COMPLETED
    EOM     =       04      ; 1 END OF MESSAGE
    EOS     =       05      ; 1 EXTERNAL INTERRUPT PENDING
    EXTIN   =       06      ; 1=EXTERNAL INTERRUP PENDING
;------------------------------------------------------------------------
    ADDR0   =       001     ; BCD TO BINARY LOCATION BINARY IN [1,0] BCD IN [3,2]
    TDBUFF  =       002     ; TOP OF DATA BUFFER HOLDS STOP ADDRESS BYTE 1
    ADDR1   =       003     ; MEMORY LOCATION FORM TEMPORARY DATA
    TEMP    =       004     ; TEMPORARY REGISTER
    TDBUF1  =       005     ; TOP OF DATA BUFFER1 HOLDS STOP ADDRESS BYTE 2
    KEYBUF  =       006     ; COMMAND BUFFER
    KEYIN   =       007     ; HOLDS ADDRESS LOWER BYTE
    KEYIN1  =       008     ; HOLDS ADDRESS UPPER BYTE
    SAVEA   =       009     ; SAVE ACUMULATOR WHEN IN INTERRUPT
    SAVEB   =       00A     ; B REGISTER SAVE AREA
    SAVEX   =       00B     ; X POINTER SAVE AREA
```

```
;026.......02F; RESERVED STACK AREA
    STACK   =       02F     ; TOP OF STACK
;   ;
.FORM
;*****************************************************************************
;       INTIALIZE REGISTERS
;       POWER UP AND PRESET ROUTINE
;*****************************************************************************
.SECT CODE,ROM,ABS=0
CLRRAM:
    LD      B,#00   ; CLEAR ALL USER  RAM including I/O ports
    LD      X,#0FC  ; AND REGISTERS
DONE:CLR    A
    X       A,[B+]
    DRSZ    X
    JP      DONE
;START PROGRAM EXECUTION
REST:LD     SP,#02F DEFAULT STACK INITIALIZATION
    LD      B,#PORTLD; CONFIGURE L-PORT
    LD      [B+],#0FF
    LD      [B],#000; MAKE ALL L PINS INPUT
    LD      B,#PORTGD; CONFIGURE G PORT
    LD      [B+],#01F; IRQ LOW, PORTG DATA REGISTER
    LD      [B],#030; SET I/O FOR G POR & Microwire
    LD      PORTD,#07; initialize portd
;*****************************************************************************
;ENABLE INTERRUPT
******************************************************************************
    LD      B,#CNTRL
    LD      [B+],#08C; CNTRL REGISTER SELECT MUIRE
    LD      [B],#003; INITIALIZE PSW
    LD      B,#0FF
;************ INITIALIZE ISD33000 ******************************
    LD      A,#000  ; INITIALIZE ADDRESS TO 00
    RBIT    SS,PORTD; ENABLE SPI
    JSR     SPIX    ; SEND OUT THE COMMAND
    LD      A,#000  ; POWER UP, USE INPUT ADDRESS REG.
    JSR     SPIX
    SBIT    SS,PORTD; DISABLE SPI
    LD      DELC1,#02; INITIALIZE DELAY COUNTER
    JSR     PWRUP   ; SEND POWER-UP COMMAND
;*****************  WAIT FOR AN INPUT HERE ******************************
;
;                    MAIN LOOP WAITING FOR A SWITCH CLOSER
;*****************************************************************************
READ:
    IFBIT   LBO,PORTGP;IF LOW BATT PIN IS LOW GO TO WARNING ROUTINE
    JP      READ1   ;ELSE PROCEED
    JMP     WARNING ;GO TO WARNING ROUTINE
******************************************************************************
;BRANCH ACCORDING TO THE KEY PUSHED ON THE KEY PAD
;*****************************************************************************
READ1:
    LD      PORTLC,#000; CONFIGURE L PORT AS INPUT PORT
    LD      PORTLD,#0FF; WEAK PULL-UP
```

```
    LD      A,PORTLP; READ PORT L
    LD      DELC1,#30; DE-BOUNCE THE SWITCH
    JSR     DELAY1
    AND     A,#0FF  ; SELECT THE  SWITCHES
    IFEQ    A,#040
    JP      PLAY    ; PLAY AT THE "NEXT" ADDRESS
    IFEQ    A,#02
    JP      REC     ; RECORD AT THE "NEXT" ADDRESS
    IFEQ    A,#010
    JP      STOP    ; STOP PLAYBACK OR RECORD
    IFEQ    A,#080
    JP      PLAY0   ; PLAY ADDRESS 0
    IFEQ    A,#008
    JP      REC0    ; RECORD AT ADDRESS 0
    IFEQ    A,#020
    JP      MSGCUE  ; DO A MESSAGE CUEING CYCLE
    IFEQ    A,#004
    JMP     GOTOB
    IFBIT   HALT,STAT; IF SLEEP MODE IS SET GO TO SLEEP ROUTINE
    JP      HALT1   ; GO TO SLEEP ROUTINE
    JP      READ    ; ELSE BACK TO MAIN
HALT1:
    RBIT    HALT,STAT; RESET SLEEP MODE FLAG
    SBIT    7,PORTGD; MICRO IS PUT TO SLEEP MODE FOR
    NOP     ; POWER SAVING
    NOP     ; THE PROGRAM RESUMES HERE WHEN A KEY
    NOP     ; RESET BUTTON IS PUSHED
    NOP
    JMP     READ    ; GO WAIT FOR AN INPUT
;*****************************************************************
;                        INITIALIZE ISD33000 Routine
;*****************************************************************
PWRUP:
    LD      A,#004  ; 00100 POWER UP ISD33000
    RBIT    SS,PORTD; SELECT ISD33000
    JSR     SPIX    ; SEND OUT THE POWER UP COMMAND
    SBIT    SS,PORTD; DISABLE SPI
    RET
;*************************************************************************
;GENERIC DELAY ROUTINE  256TC X 256TC( TC= INSTRUCTION CYCLE )
;*************************************************************************
DELAY1:LD   DELC2,#0FF
DELAY2:DRSZ DELC2   ; DELAY ROUTINE MAX LENGTH 256 LOOPS
    JP      DELAY2
    DRSZ    DELC1
    JP      DELAY1
    RET
;*********************************************************
                ;MESSAGE CUEING CYCLE
;*********************************************************
MSGCUE:
    SBIT    LED,PORTD; FLASH LED FOR 50MS TO INDICATE
    ; A MSG CUEING CYCLE
    LD      DELC1,#0F; DE-BOUNCE THE SWITCH
    JSR     DELAY1  ; USE DELAY1 ROUTINE
```

```
    RBIT    LED,PORTD; LED OFF
    LD      A,#01F  ; SEND MESSAGE CUEING COMMAND ;
    ; WITH IAB = 1
    RBIT    SS,PORTD; ENABLE SPI
    JSR     SPIX    ; SEND OUT THE COMMAND
    SBIT    SS,PORTD; DISABLE SPI
    JMP     READ    ; DONE GO TO MAIN ROUTINE
.FORM
*****************************************************************************
    *BEGIN INTERRUPT ROUTINE
 .=X'00FF
;*****************************************************************************
;INTERRUPT SERVICE ROUTINE , THE PROGRAM BRANCHES HERE WHEN THE PROCESSOR IS INTERRUPTED
;*****************************************************************************
INT1:
    IFBIT   IPND,PSW; CHECK  IF EXTERNAL INTERRUPT PENDING
    JP      EXT     ; GO SERVICE EXT. INTERRUPT
    JMP     CLRRAM  ; ELSE ILLEGAL CONDITION  RESET PROCESSOR
EXT:
    JSR     PUSH    ; SAVE REGISTERS
    RBIT    IPND,PSW; RESET EXTERNAL INTERRUPT PENDING
    JSR     STOPX   ; ISSUE A STOP COMMAND IF END OF MESSAGE
    IFBIT   07,TDBUFF; CHECK IF OVERFLOW ?
    JSR     OVFPO   ; YES GO TO OVERFLOW RECOVERY ROUTINE
    JSR     POP     ; RETURN FROM INTERRUPT
    RETI
;********************* END INTERRUPT ROUTINE*************************
*****************************************************************************
;RECORD AT "LAST" ADDRESS
;*****************************************************************************
REC:
    IFBIT   BEG,STAT; IF BEG OF MEMORY GO RECORD FROM 0
    JMP     REC0\   ; ELSE
    LD      A,#000  ; READ THE ADDRESS POINTER
    RBIT    SS,PORTD        ;
    JSR     SPIX    ;
    LD      A,KEYBUF        ; READ THE OUTPUT DATA
    X       A,KEYIN         ; SAVE IT IN KEY IN
    LD      A,#00D          ;
    JSR     SPIX    ; GET THE DATA FROM THE INCOMING BUFFER
    LD      A,KEYBUF; READ THE OUTPUT DATA
    X       A,KEYIN1; SAVE SECOND BYTE
    SBIT    SS,PORTD;
    SBIT    NEW,STAT        ; INDICATE A NEW RECORDING CYCLE
    RBIT    D2,PORTD        ; POWER AMPS ON
    RBIT    D1,PORTD        ; RECORD MODE
    SBIT    LED,PORTD       ; REC OPERATION IN PROGRESS
    JMP     READ    ; DONE WAIT FOR ANOTHER COMMAND OR OVF
;******************** SPI ROUTINE*********************************************
;
;THIS ROUTINE SENDS ANY GIVEN COMMAND AND RETURNS THE ADDRESS
;POINTER ONE BYTE AT THE TIME SAVES THE RETURN ADDRESS IN KEYBUF
;BE AWARE THAT ANY UN READ KEYBUF WILL BE LOST WITH THE NEXT SPI
;COMMAND
;*****************************************************************************
;
```

```
SPIX:
    X       A,SIOR  ; LOAD THE SPI SHIFT REGISTER
    SBIT    #BUSY,PSW; ENABLE THE SPI SHIFT REGISTER
WAIT:IFBIT #BUSY,PSW; IF IN THE MIDDLE OF TRANSMISSION
    JP      WAIT    ; WAIT until done
    X       A,SIOR  ; READ SPI SHIFT REGISTER
    X       A,KEYBUF; SAVE THE STATUS IN KEYBUF
    RET
;*****************************************************************
; STOP WITH IAB=1
STOP:JSRSTOPX; SEND A STOP COMMAND
    JMP     READ
;*****************************************************************
;                   STOP ROUTINE
;*****************************************************************
STOPX:RBIT LED,PORTD; LED OFF
    SBIT    D2,PORTD; DISABLE POWER AMPS
    LD      A,#000  ; SET THE STOP COMMAND WITH POWER UP BIT
    RBIT    SS,PORTD; DISABLE SPI
    JSR     SPIX
    LD      A,KEYBUF; GET THE ADDRESS FROM THE BUFFER
    X       A,TDBUFF; SAVE IT IN TDBUFF MEMORY LOCATION
    LD      A,#00E  ; SET THE STOP COMMAND WITH POWER UP BIT
    ; IN PLAY MODE WITH IAB = 1
    JSR     SPIX    ;
    LD      A,KEYBUF;
    X       A,TDBUF1; SAVE THE 2ND BYTE
    SBIT    SS,PORTD; DISABLE SPI
    SBIT    HALT,STAT; GO TO SLEEP MODE
    RET
;*****************************************************************
;   IGNORE ADDRESS BITS
;   ; PLAY AT "LAST" ADDRESS   ROUTINE
;*****************************************************************
PLAY:
    RBIT    D2,PORTD
    SBIT    D1,PORTD; UT ISD1000A IN PLAY MODE
    SBIT    LED,PORTD; INDICATE PLAY IN PROGRESS
    IFBIT   BEG,STAT; CHECK IF PLAY FROM BEG OF MEMORY
    JMP     PLAY2   ; YES, RESET ADDRESS TO 00 AND PLAY
PLAYN:
    LD      A,#00E  ; PLAY IAB = 1, PU =1, P/R =1 RUN =0
    RBIT    SS,PORTD;
    JSR     SPIX    ;
    SBIT     SS,PORTD;
    LD      A,#00F  ; PLAY IAB = 1, PU =1, P/R =1 RUN =1
    RBIT    SS,PORTD;
    JSR     SPIX    ;
    SBIT    SS,PORTD;
    JMP     READ    ; DONE GO WAIT FOR ANOTHER COMMAND
PLAY2:LD    A,#000  ; PLAY IAB = 1, PU =1, P/R =1 RUN =0
    RBIT    SS,PORTD;
    JSR     SPIX    ;
    LD      A,#007  ; PLAY IAB = 1, PU =1, P/R =1 RUN =1
    RBIT    SS,PORTD;
```

```
    JSR     SPIX    ;
    SBIT    SS,PORTD;
    RBIT    BEG,STAT; RESET BEG OF MEMORY FLAG
    JMP     PLAYN   ; GO CONTINUE
;*********************************************************************
;GO TO BEGINNING OF MEMORY  THIS ROUTINE ONLY FLASHES AN LED AND SETS
; A FLAG TO INDICATE THE BEG OF MEMORY BUTTON IS PUSHED
;*********************************************************************
GOTOB:
    SBIT    LED,PORTD; LED ON
    SBIT    BEG, STAT;
    LD      DELC1,#0F; DE-BOUNCE THE SWITCH
    JSR     DELAY1
    RBIT    LED,PORTD; LED OFF
    LD      DELC1,#0F; DELAY
    JSR     DELAY1  ;
    SBIT    LED,PORTD; LED ON
    LD      DELC1,#0F; DELAY
    JSR     DELAY1  ;
    RBIT    LED,PORTD; LED OFF
    JMP     READ    ; DONE WAIT FOR ANOTHER COMMAND
;*********************************************************************
;START PLAY AT ADDRESS 0 IGNORE ADDRESS BITS
;*********************************************************************
PLAY0:JSR   PLAY1   ; USE THE PLAY1 ROUTINE
    JMP     READ    ; DONE GO WAIT FOR ANOTHER COMMAND
PLAY1:
    RBIT    D2,PORTD; PD PIN TO 0
    SBIT    D1,PORTD; PUT ISD1000A IN PLAY MODE (P/R)
    SBIT    LED,PORTD;
    IFBIT   NEW,STAT; IF THERE IS NO NEW RECORD SKIP
    JMP     PLAYNW  ;
    JMP     PLAYA   ; PLAYA  ELSE
PLAYNW:
    LD      TEMP,#00; THIS ROUTINE READS THE ADDRESS POINTER
    IFBIT   5,KEYIN ; PREPARES IT FOR SPI REGISTER
    SBIT    7,TEMP  ; SAVE THE FISRT BYTE IN KEYIN AND SECOND
    IFBIT   4,KEYIN ; BYTE IN KEYIN1 MEMORY LOCATIONS
    SBIT    6,TEMP
    IFBIT   3,KEYIN
    SBIT    5,TEMP
    IFBIT   2,KEYIN
    SBIT    4,TEMP
    IFBIT   1,KEYIN
    SBIT    3,TEMP
    IFBIT   0,KEYIN
    SBIT    2,TEMP
    LD      A,TEMP
    X       A,KEYIN
    LD      A,KEYIN1
    X       A,TEMP
    IFBIT   7,TEMP
    SBIT    1,KEYIN
    IFBIT   6,TEMP
    SBIT    0,KEYIN
```

```
    LD      A,#00
    X       A,KEYIN1
    IFBIT   5,TEMP
    SBIT    7,KEYIN1
    IFBIT   4,TEMP
    SBIT    6,KEYIN1
    RBIT    NEW,STAT
PLAYA:
    LD      A, KEYIN; READ THE KEYIN BUFFER
    RBIT    SS,PORTD; SEND OUT THE ADDRESS
    JSR     SPIX
    LD      A,#007  ; PLAY IAB = 0, PU =1, P/R =1 RUN =1
    OR      A, KEYIN1;
    JSR     SPIX    ;
    SBIT    SS,PORTD; DISABLE SPI
    RBIT    SS,PORTD
    LD      A,#00E  ; PLAY IAB = 1, PU =1, P/R =1 RUN =0
    RBIT    SS,PORTD
    JSR     SPIX    ;
    SBIT    SS,PORTD;
    RBIT    SS,PORTD;
    LD      A,#00F  ; PLAY IAB = 1, PU =1, P/R =1 RUN =1
    RBIT    SS,PORTD;
    JSR     SPIX    ;
    SBIT    SS,PORTD;
    RET     ;
;*********************************************************************
                ;RECORD AT ADDRESS 0
;*********************************************************************
REC0:
    RBIT    D2,PORTD; POWER AMPS OFF
    RBIT    D1,PORTD; RECORD MODE
    SBIT    LED,PORTD
    LD      A,#000  ; INITIALIZE ROWS =000
    RBIT    SS,PORTD; ENABLE SPI
    JSR     SPIX    ;
    LD      A,#005  ; REC IAB = 0, PU = 1, P/R =0 RUN =1
    JSR     SPIX    ;
    SBIT    SS,PORTD; DISABLE SPI
    LD      A,#00D  ; REC IAB = 1, PU = 1, P/R =0 RUN =1
    RBIT    SS,PORTD;
    JSR     SPIX    ;
    SBIT    SS,PORTD;
    RBIT    BEG,STAT;
    JMP     READ
;*********************************************************************
;PUSH AND POP ROUTINES FOR SAVING REG. DURING INTERRUPT
**********************************************************************
POP:
    LD      A,SAVEX
    X       A,X
    LD      A,SAVEB
    X       A,B
    LD      A,LCDPTR
    X       A,PORTLD
```

```
    LD      A,SAVEA
    RET
;
PUSH:
    X       A,SAVEA
    LD      A,B
    X       A,SAVEB
    LD      A,X
    X       A,SAVEX
    LD      A,PORTLD
    X       A,LCDPTR
    RET
WARNING:
    SBIT    LED,PORTD
    LD      DELC1,#50; DE-BOUNCE THE SWITCH
    JSR     DELAY1
    RBIT    LED,PORTD
    LD      DELC1,#50; DE-BOUNCE THE SWITCH
    JSR     DELAY1
    JP      WARNING
;*********************************************************************
;                   OVERFLOW RECOVERY ROUTINE
; INITIALIZE ADDRESS POINTERS TO 00 IN PLAY MODE AND IMMEDIATELY STOP THE OPERATION.
;*********************************************************************
OVFPO:
    RBIT    SS,PORTD
    LD      A,#000  ; POWER UP, USE INPUT ADDRESS REG.
    JSR     SPIX    ;
    LD      A,#007  ; PLAY MODE WITH IAB = 0
    JSR     SPIX
    SBIT    SS,PORTD
    LD      A,#000  ; POWER UP, USE INPUT ADDRESS REG.
    RBIT    SS,PORTD
    JSR     SPIX
    LD      A,#006  ; STOP MODE WITH IAB = 0
    JSR     SPIX
    SBIT    SS,PORTD
    RET
 .ENDCLRRAM
;******************END OF PROGRAMM*********************************
```

INSTRUCTIONS

Six of the 9 push-buttons are labeled on the PCB. Pushing the yellow "GO_TO_BEG" button will reset the address pointer to the front of the chip or address 000. There will be a double flash of the LED, D11, in the bottom left corner of the PCB to indicate that this has been done.

Pushing the red "RECORD" button will turn on the red LED to indicate that the chip is now recording anything it hears at the microphone M1 near the top center of the board. The board will continue to record until the end of the chip is reached or the black "STOP" button is pressed. At that time, the LED will go out and the board will stop recording.

Pushing the white "PLAY_LAST" button will playback what was just recorded. The message will play through to its end or stop when the black "STOP" button is pressed.

Pushing the yellow "GO_TO_BEG" button and then the white "PLAY_NEXT" button will play messages from the beginning of memory through to the end or stop/pause when the black "STOP" button is pressed. To resume playback after pause, push PLAY_NEXT again. To play the next message, press PLAY_NEXT.

The blue "SKIP_TO_NEXT" button bypasses a message and plays the one after. For example, if three message are recorded beginning at the front of the chip, after the last message press the white "PLAY_NEXT" button to play the first message, press the blue "SKIP_TO_NEXT" button to bypass the second message and then press the white "PLAY_NEXT" to play the third message.