



---

# ATMEL AVR 2004 DESIGN CONTEST

---



A NAVIGATIONAL AID FOR THE VISUALLY IMPAIRED

**PROJECT NO : A3560**

## PROJECT SUMMARY ( 500 WORDS ) :

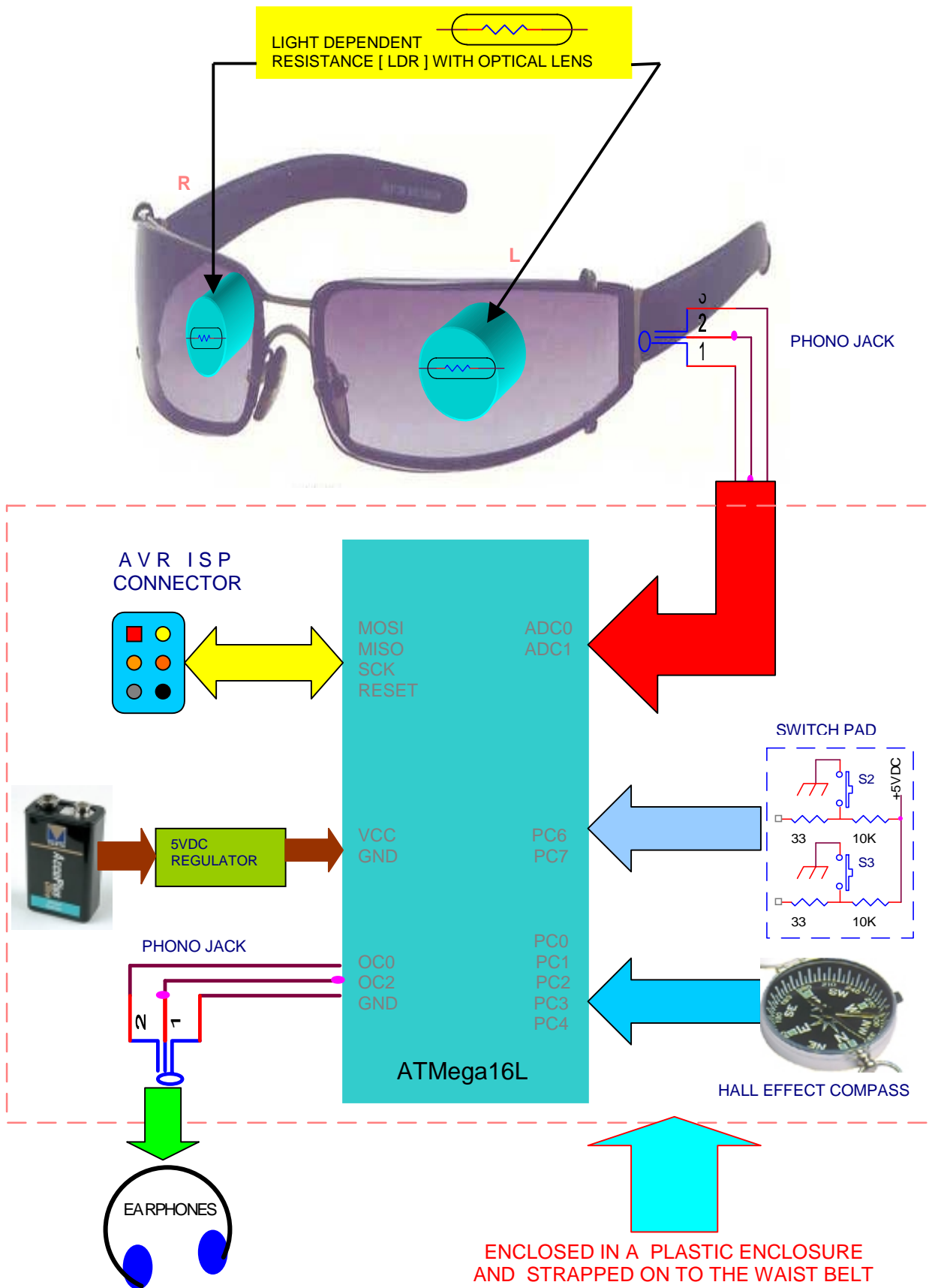
The Navigational Aid for the visually impaired ( acronymed NAVI ) intended primarily for use by people with vision impairment, is a compact gadget that gives a sense of ambient light and geographic direction through auditory representation of varying audio tones and Morse coded messages over a stereo channel. The aid is based on the ATMEL, ATmega16 Flash Microcontroller which forms the heart of this gadget. No visual displays are provided except for a pair of light-glasses and earphones through which light and direction data are collected and communicated to the user. Three conveniently placed pushbuttons are used for operating the system. The electronic circuit board comprises of the ATmega16, some passive components and a Hall Effect compass housed in a small unit that runs from a 9 volt cell and may be strapped on to a waist belt with the earphones and light-glasses worn during use.

The magic inside the black box is pretty simple. A pair of lightglasses based on two large light dependent resistors (LDR's) collect information on ambient light and their transient variations and feed the same into the analog port of the ATmega16. The microcontroller scans the analog pins for light data and translate the the same into audio tones starting from 800 Hz (approx) to 2000 Hz from a dark to bright hierarchy. The total darkness or very dark is a no tone state and from the lowest threshold, the tone starts with a gradual increase to the highest pitch on a very strong light signal. The tones are effectively generated by the two system on chip timers and a stereographic vision of the ambient light over the left and right eyes are communicated to the user. A pair of optical lenses capped on to the LDR's focuses the incoming light.

There's still more. The black box contains a hall effect compass with four sensors placed at right angles to detect the position of the magnetic needle and ascertain the geographic directions of the four cardinal points N - E - S - W. The user has to simply move in a clockwise rotation to lock on to the first cardinal point either North, East, South or West. The hall effect sensings are detected by a port on the ATmega16 and the respective direction is communicated to the user by a morse coded message over a string of dits and dahs at a frequency of 1000 Hz.

The unit with its embedded firmware and a handful of active & passive devices runs from a 9VDC battery. On startup the vision menu is invoked and the user may switch back and forth from the vision to the compass menu by pressing a pair of push buttons that are located on top of the unit. The unit is meant to worn along with a waist belt and should be kept in a straight position for the compass to work correctly. The operation of this gadget is simple and and requires no special skills whatsoever.

**SYSTEM BLOCK DIAGRAM :**



## TECHNICAL SPECIFICATIONS :

### **A. LIGHT GLASSES :**

Parameter	Units	Specifications
Type of Sensing	-	Binocular over both the eyes
Area of sensing	mm	Over a sensor dia of 20 mm
Range of sensing	ft	Over a min. range of approx. 6 ft
Mode of sensing	-	Average light intensity
Sensor types	-	Light Dependent Resistance
Sensor specifications (DARK)	ohm	Full Dark > 120 kohm
Sensor specifications (BRIGHT)	ohm	Very Bright < 500 ohm
Sensor attachments	No.	2 nos. convex lens
Sensor holding frame	No.	Side covered sunglasses
Sensor output port	-	Stereo phono jack with chord

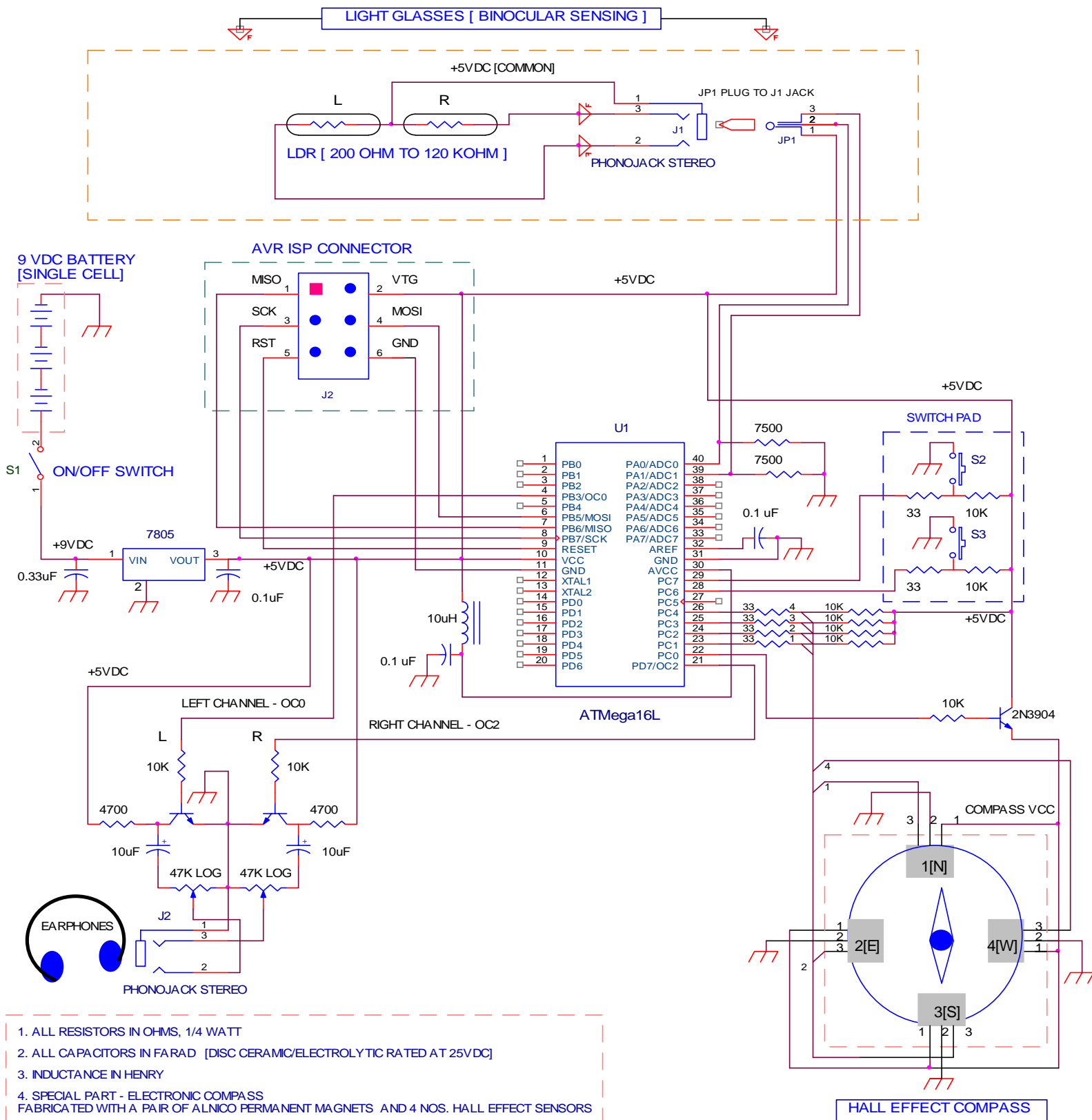
### **B. ELECTRONIC COMPASS :**

Parameter	Units	Specifications
Compass Mechanism	-	High carbon steel pivot with MS strip indicator & pellet magnets(2)
Compass Enclosure	-	Nylon enclosure with plexi top lid
Compass Enclosure dimensions	mm	26mm dia, height 27mm
Compass pivot/fulcrum height	mm	Tapered pivot of height 5 mm
Compass Indicator dimensions	mm	Approx. length of 12 mm
Compass Magnet dimensions	mm	2 mm dia; 1.5 mm thick, pellets
Compass Magnet placement	-	At indicator tip in N-S orientation
Compass Sensors	No.	4 nos. Hall Effect sensors
Compass arrangement	degree	placed at 90° from one another
Hall Device Sensor distance	mm	A max. of 3 mm from indicator tip
Hall Device Supply	VDC	Max. of 5 VDC
Compass Position	degree	Vertical; a max tilt of 10 degrees

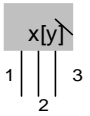
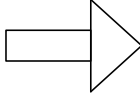
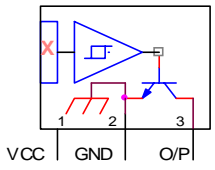
### **C. ATMEL MICROCONTROLLER UNIT :**

Parameter	Units	Specifications
Analog Input	No.	2 inputs with max. limit of 5 VDC
Analog Input resolution	mV	8 bit ADC with approx. 20 mV res.
Analog Ref. voltage	VDC	5 VDC
Digital Inputs (Switch)	No.	2 nos. Command Execute Keys
Digital Inputs (Sensors)	No.	4 nos. Hall Effect Sensors
Digital Outputs	No.	1 no. for Compass Drive
Timer Outputs	No.	2 nos for L/R channels (OC0 & 2)
Timer Output (LIGHT data)	Hz	800 to 2000 Hz (Dark to Bright)
Timer Output (Direction data)	Hz	Approx. 1000 Hz, morse encoded
Morse audio symbols	-	Standard Morse Code (dit/dah)
Interrupt Type	-	Polled Interrupts
Microcontroller Supply	VDC	5 VDC derived from 9VDC

# ELECTRONIC CIRCUIT SCHEMATIC :



1. ALL RESISTORS IN OHMS, 1/4 WATT
2. ALL CAPACITORS IN FARAD [DISC CERAMIC/ELECTROLYTIC RATED AT 25VDC]
3. INDUCTANCE IN HENRY
4. SPECIAL PART - ELECTRONIC COMPASS  
FABRICATED WITH A PAIR OF ALNICO PERMANENT MAGNETS AND 4 NOS. HALL EFFECT SENSORS
5. S1 - SPDT SWITCH, S2&S3 - MIN. PUSHBUTTON



HALL EFFECT SENSOR  
-3144/3175 FROM  
ALLEGRO  
MICROSYSTEMS INC  
USA.

ATMEL AVR 2004 DESIGN CONTEST			
Title	A NAVIGATIONAL AID FOR THE VISUALLY IMPAIRED		
Size	Document Number	PROJECT NO. A3560	Rev 0
Date:	Monday, May 31, 2004	Sheet 1	of 1

```
*****
;*
;*          ATMEL AVR 2004 DESIGN CONTEST          *
;*
;*    A NAVIGATIONAL AID FOR THE VISUALLY IMPAIRED ( NAVI )    *
;*
;*          PROJECT NO. A3560          *
;*
;*  ATMega16L running at 2 MHz; Osc. Calibration Byte:0x1A      *
;*
;*
*****
```

```
.include "m16def.inc" ;Includes the ATMega16 definitions file
```

```
*****
; REGISTER DEFINITIONS & NAMES          *
*****
```

```
.def Temp=r16
.def Temp_1=r17
.def TOP0=r18
.def TOP1=r19
.def ADC_Value=r20
.def VF_Value=r21
.def Mux_Value=r22
.def Outer_L=r23
.def Inner_L=r24
.def CP_Value=r25
```

```
*****
; INTERRUPT VECTORS          *
*****
```

```
.org 0x0000          ;Places the following code from address 0x0000
rjmp RESET          ;Take a Jump to the RESET Label
```

```
*****
; INITIALISATION          *
*****
```

```
RESET:              ;Reset Label / Initialisation
```

```
ldi Temp, HIGH(RAMEND) ;Configure Stackpointer
out SPH, Temp          ;
ldi Temp, LOW(RAMEND)
out SPL, Temp
```

```
ldi Temp, 0x1A        ; Calibrate Internal Oscillator at 2MHz
out OSCCAL, Temp      ; Adjusted at 1A for best stability
```

```
*****
; CONFIGURE PORTS
*****
```

```
Setport:           ; Port configuration
```

```

sbi DDRB, 3          ; Set DDRB, Bit3 as output for OC0/Waveform
                    ; output on OC0-Toggle pin on compare match
sbi DDRD, 7          ; Set DDRD, Bit7 as output for OC2/Waveform
                    ; output on OC2-Toggle pin on compare match
sbi DDRC, 0          ; Set DDRC, Bit 0 to switch on the compass

;*****
;          CONFIGURE ANALOG TO DIGITAL CONVERSION SYSTEM ON CHIP          *
;*****
;*
;* ADMUX - ADC Multiplexer Selection Register                               *
;*
;* Bit 7:6 (REFS1:0) Voltage Reference Selection Bits                       *
;* value: 01 -> AVCC with external capacitor at AREF pin                    *
;*
;* Bit 5 (ADLAR) ADC Left Adjust Result                                    *
;* value: 1 -> Left adjust selected ( 8 bit ADC value used )              *
;*
;* Bit 4:0 (MUX 4:0) Analog Channel and Gain Selection Bits                *
;* value: 00000 -> First, ADC0 is selected as single ended input          *
;*****

    ldi Mux_Value, 0b01100000 ; Set ADC parameters
    out ADMUX, Mux_Value      ; Load in ADMUX register
    sbi ADCSR, ADEN           ; Enable the ADC module
    rjmp S0                    ; First start instruction

;*****
; MAIN PROGRAM STARTS...START AD CONVERSION ON ADC00, FETCH VALUE *
; COMPARE TO TABLE AND CONVERT VOLTAGE VALUE TO FREQUENCY VALUE.. *
; TIMER0, OCR0, TOP VALUE SET AS PER VF VALUE (FREQUENCY VALUE) *
;*****

Start:                                ; Main program loop starts here

    cbi PORTC,0                 ; PC0 is driven low to switchoff compass module
    sbis PINC, 7                ; SW2, Low when compas func. selected
    jmp COMPASS                 ; Jump to Compass routine

    cbi ADMUX, 0                ; Set ADC on channel ADC0
S0: ldi VF_Value, 0x00          ; Highest Pitch/Timer0 frequency
    ldi CP_Value, 0xFF          ; Maximum Light/ADC Value

ADC0_Start:
    sbi ADCSR, ADSC             ; Start AD conversion

ADC0_Loop:
    sbis ADCSR, ADSC           ; Poll ADSC bit, if 0 then conversion complete
    jmp ReadValue0_Compare
    jmp ADC0_Loop

ReadValue0_Compare:

    in ADC_Value, ADCH          ; Fetch 8 bit ADC value
    cpi ADC_Value, 0x0A         ; Check for min signal
    brlo Dark0                 ; branch to Dark Signal or NO LIGHT

CP0_Loop:

```

```
    cp    ADC_Value, CP_Value
    brsh  Freq_Select_Timer0
    inc   VF_Value
    dec   CP_Value
    rjmp  CP0_Loop

Freq_Select_Timer0:

    mov   TOP0, VF_Value
    ldi   Temp, 0x1A      ; All set for Timer0 triggering
    rjmp  ADC1           ; ADC0 cycle complete; go to ADC1

Dark0:                                ; Dark Signal or NO LIGHT

    ldi   Temp, 0x0C      ; load value for OC0 pin disconnection
    rjmp  ADC1           ; ADC0 cycle complete; go to ADC1

;*****
;Start ADC on channel ADC1,change ADMUX bit0 and repeat prcedure *
;*****

ADC1:

    sbi   ADMUX, 0        ; Set ADC on channel ADC1
    ldi   VF_Value, 0x00  ; Highest Pitch/Timer2 frequency
    ldi   CP_Value, 0xFF  ; Maximum Light/ADC Value

ADC1_Start:
    sbi   ADCSR, ADSC     ; Start AD conversion

ADC1_Loop:
    sbis  ADCSR, ADSC     ; Poll ADSC bit,if 0 then conversion complete
    jmp   ReadValue1_Compare
    jmp   ADC1_Loop

ReadValue1_Compare:

    in    ADC_Value, ADCH  ; Fetch 8 bit ADC value
    cpi   ADC_Value, 0x0A  ; Check for min signal
    brlo  Dark1           ; branch to Dark Signal or NO LIGHT

CP1_Loop:

    cp    ADC_Value, CP_Value
    brsh  Freq_Select_Timer2
    inc   VF_Value
    dec   CP_Value
    rjmp  CP1_Loop

Freq_Select_Timer2:

    mov   TOP1, VF_Value
    ldi   Temp_1, 0x1A     ; All set for Timer2 triggering
    rjmp  TIMER0_2        ; Now jump to Timer schedule

Dark1:                                ; Dark Signal or NO LIGHT

    ldi   Temp_1, 0x0C     ; load value for OC2 pin disconnection
```

```
                ; ADC0 & ADC1 cycles complete

;*****
; TIMER0 & 2 ACTIVATION - TIMER COMPARE VALUE TAKEN FROM TOP VALUE INPUT
;*****

Timer0_2:      ; Start of Timer schedule - all req. values loaded
                ; in the previous steps; Now just load to Output
                ; Compare & Timer Control registers to start the
                ; the timers toggling OC0 & OC2 pins or otherwise
                ; disconnect the OC0 & OC2 pins for dark signals

                out  OCR0, TOP0    ; TOP0 value (VF value) loaded into Timer0
                out  OCR2, TOP1    ; TOP1 value (VF value) loaded into Timer2

                out  TCCR0, Temp   ; Set Timers for CTC,with output toggle OC0 & OC2
                out  TCCR2, Temp_1 ; or 0x0C for disconnection. Normal Code - 0x1A for
                ; toggle output on OC0 & OC2, CTC mode and
                ; Prescale clock_I/O by /8

                call DelayA
                rjmp Start        ; Keep looping - refresh Timer0 & 2 with VF Value

;*****
; Compass Routine - Detection of Low signals on the Hall Effect pins on
; PortC at PC1, PC2, PC3, PC4 and playing the morse coded message
;*****

COMPASS:

                sbis PINC, 6       ; SW3, Low when return from compas func. selected
                jmp  START        ; jump back to START
                ldi  Temp, 0x0C   ; Prepare to cutoff timers from ADC mode
                out  TCCR0,Temp    ; cutoff timer0
                out  TCCR2,Temp    ; cutoff timer2
                sbi  PORTC,0       ; PC0 is driven high to activate compass module
                sbis PINC, 1       ; Test Bit 1/PC1
                jmp  North
                sbis PINC, 2       ; Test Bit 2/PC2
                jmp  East
                sbis PINC, 3       ; Test Bit 3/PC3
                jmp  South
                sbis PINC, 4       ; Test Bit 4/PC4
                jmp  West
                jmp  Compass      ; If all bits high, start again

North:

                call Dah
                call Dit
                call DelayA
                call DelayA
                call DelayA
                jmp  COMPASS

East:

                call Dit
                call DelayA
```

```
call DelayA
call DelayA
jmp COMPASS
```

South:

```
call Dit
call Dit
call Dit
call DelayA
call DelayA
call DelayA
jmp COMPASS
```

West:

```
call Dit
call Dah
call Dah
call DelayA
call DelayA
call DelayA
jmp COMPASS
```

```
;*****
; DELAY SUBROUTINE
;*****
```

Delay:

```
ldi Outer_L, 0x00
```

Delay00:

```
ldi Inner_L, 0x00
```

Delay01:

```
nop
inc Inner_L
cpi Inner_L, 0xFF
brne Delay01
inc Outer_L
cpi Outer_L, 0xFF
brne Delay00
ret
```

```
;*****
; DELAY SUBROUTINE_SHORT
;*****
```

DelayA:

```
ldi Outer_L, 0x00
```

DelayA0:

```
ldi Inner_L, 0x00
```

DelayA1:

```
nop
inc Inner_L
cpi Inner_L, 0xFF
brne DelayA1
inc Outer_L
cpi Outer_L, 0x50
brne DelayA0
ret
```

```
*****
; Subroutines for generation of Morse Code, dit & dah and Fixed Tone
*****
```

Dah:

```
call Timer02_Fixed
call DelayA
call DelayA
call DelayA ; x3
ldi Temp, 0x0C ; shut off
out TCCR0, Temp ; completes dah
out TCCR2, Temp ; completes dah
call DelayA ; one dit gap - no sound
ret
```

Dit:

```
call Timer02_Fixed ;
call DelayA ; x1
ldi Temp, 0x0C ; shut off
out TCCR0, Temp ; completes dit
out TCCR2, Temp ; completes dit
call DelayA ; one dit gap - no sound
ret
```

Timer02\_Fixed: ; used for the morse code generation

```
ldi Temp, 0xAA
out OCR0, Temp ; TOP value (VF value) loaded into Timer0
out OCR2, Temp ; TOP value (VF value) loaded into Timer2
ldi Temp, 0x1A ;
out TCCR0, Temp ; Configure counter for CTC, with output toggle OC0
out TCCR2, Temp ; Configure counter for CTC, with output toggle OC2
ret
```

```
*****
; END OF PROGRAM
*****
```