Flux Gate Musical Toy

FGM-3 Flux Gate Toy . . . . .

While this could be classed as a toy, it's also a very sensitive magnetic sensing project which has many other applications. The "toy" idea came up from the idea that kids can enjoy the mystery of magnetism by waving a magnet a few feet away from the sensor to change the pitch of a tone.

Dancing with the magnet, then, would yield their own music.

The ATTiny2313 micro-controller was chosen to enhance this project. By doing so it allowed a frequency division of the FGM-3 sensor output .... or a beating against a crystal controlled frequency standard to enable high sensitivity.

A single push button selects one of several options:

1. RESET
2. DIVIDE THE FLUX GATE OUTPUT BY 7
3. DIVIDE THE FLUX GATE OUTPUT BY 8
4. DIVIDE THE FLUX GATE OUTPUT BY 9
5. DIVIDE THE FLUX GATE OUTPUT BY 10
6. HIGH SENSITIVITY, BEAT FREQUENCY, OPTION.
7. RETRIEVE EEPROM SAVED BEAT FREQUENCY
8. SAVE BEAT FREQUENCY IN EEPROM

When using option 6, you have to wait several seconds for it to find the correct beat frequency to achieve good audio listening. Once found, you can tweak the position of the sensor to fine tune the desired audio pitch.

Also, one can save a beat frequency currently being used to EEPROM (#8). And, a saved beat frequency in EEPROM can be loaded when desired (#7).

Additionally, the following options are available for additional effects:

12. TOGGLE THE 7TH DIVISION WITH PRESENT TONE
13. TOGGLE THE 8TH DIVISION WITH PRESENT TONE
14. TOGGLE THE 9TH DIVISION WITH PRESENT TONE
15. TOGGLE THE 10TH DIVISION WITH PRESENT TONE
16. TOGGLE THE HIGH SENSITIVITY TONE WITH PRESENT TONE

Description of Project:

Simply dividing the FGM-3 frequency down by 7 thru 10 dividers was the first goal. The CD4040 CMOS counter was used. This provides low sensitivity use.

For high sensititivity, a beat frequency approach was used. By having a stable crystal referenced signal beat against the FGM-3 flux gate frequency,

one could generate a tone in the audio frequency range. A CD4013 bistable circuit was used to act as the "mixer." The micro-controller started searching at the highest beat It then stepped down periodically for the next lower frequency test.

A resistance/capacitance circuit monitors the output of the CD4013, pin 13. When the frequency is within audio frequency range, the micro-controller stops its frequency search and locks onto that last beat frequency.

A 10 MHz crystal is used for the ATTiny2313. Brown out detection was set for 4.7 volts. A double filtered power supply using an 8 volt and 5 volt 78L05/08 IC was used.

For licensing:
Wayne Simister
Automated Sound
April 22, 2011
wsimister@hotmail.com

~~~~~~

```
;Flux Gate Toy
;April  25, 2011

;Project includes TWO modes:  (1) a divide down of the flux gate frequency and (2) a beat frequency a  gainst
;  the frequency generated by this program.  Either will provide listenable audio frequency.

;It is necessary to set the frequency to beat against the flux gate signal.  It must be lower than the flux
;  gate frequency.

;FLAGS

;8-  MHz - 14Clk., Sel.=1101, SUT11
;  Do not use divide by 8

;The following dit commands are used:

;  1 = Clear all tones
;  2 = Use 7th division for flux gate signal
;  3 = Use 8th division for flux gate signal
;  4 = Use 9th division for flux gate signal
;  5 = Use 10th division for flux gate signal
;  6 = Use Beat Frequency
;  7 = Use Previous Beat Frequency in EEPROM
;  8 = SAVE current Beat Frequency to EEPROM
;  9 =
;  10 =
;  12 = Toggle on/off 7th division for flux gate signal
;  13 = Toggle on/off 8th division for flux gate signal
;  14 = Toggle on/off 9th division for flux gate signal
;  15 = Toggle on/off 10th division for flux gate signal


.include "tn2313def.inc"

; R30,31 ijmp register - determines where to jump for variable frequency

; R16 general purpose temporary use register
; R17 general purpose temporary use register

; R24,R25,R26,R27,R28,R29,= scratch
```
~~~~~~

```
; R18 is flags register
;		R18,0(1) = Enable 7th
;		R!8,1(2) = Enable 8th
;		R18,2(4) = Enable 9th
;		R18,3(8) = Enable 10th
;		R18,4(16) = Enable Beat Frequency
;		R18,5(32) = STOP flag
;		R18,6(64) = 4013-23 monitor
;		R18,7(128) =

;		r26 = Button Low Byte of total count
;	r27 = Button High Byte of total count

;		r30,31 used for ijmp to delay ladder 0,17 yields lowest freq.  1,50 yields highest freq.


;*****Initialization


START1:				;set up stack pointer and initialize cold start

ldi	r16,low(RAMEND)	; Load low byte address of end of RAM into register R16
out		SPL,R16			; Initialize stack pointer to end of internal RAM

ser		r16			;Set to all ones for out port direction as output
out		DDRB,r16	;Set PORTB to output

clr		r16
out		PortB,r16	;LEDs are off when port pins are low

ldi		R16,0		;Set port D, bit 0(1), 1(2), 2(4), 3(8), 4(16), 5(32), 6(64) as inputs
out		DDRD,R16	;execute data direction

clr		r16
out		PORTD,R16	;execute to set out port D1(2) low, and remaining D input ports with
				; no input resistor

clr		r18				;zero all flags in register r18
clr		r19				;zero all flags in register r19

rcall	DelayTwo		;give 5 second delay to allow button circuit to charge

rcall	Initialize1

nop					;To assure start of ijmp at 16 or 145 lines in


;-----------------------------------

ijmp	;FrequencyDivisionTree

;Subroutine for frequency creation

FrequencyDivisionTree:


nop
nop
nop
nop
nop
nop
nop
nop
nop
nop			;ten

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop			;twenty
```

```
nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;thirty

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;forty

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;fifty

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;sixty

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;seventy

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;eighty

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;ninety
```

```
nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;one hundred

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;one hundred ten

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;one hundred twenty

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;one hundred thirty

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;140

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;150

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;160
```

```
nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;170

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;180

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;190

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;200

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;210

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;220

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;230
```

```
nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;240

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;250

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;260

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;270

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;280

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop                                     ;290


;Start of data section


sbis            PIND,0                  ;skip if PIND,0(1) is set - switch is up
rjmp            ServiceSwitch


;Start of activate output from divider, 4040 CMOS chip

Enable1:                                ;7th Enable
```

```
sbrs    r18,0                   ;skip if r18,0(1) is set
rjmp    Enable1c3               ; out -- no action

sbis    PIND,2                  ;skip if PIND2(4) is set - divide by 7
rjmp    Enable1b

sbi             PortB,1                 ;set PortB,1(2)
rjmp    Enable1c7


Enable1b:

cbi             PortB,1                 ;clear PortB,1
rjmp    Enable1c8


Enable1c3:

push    r0
pop             r0                              ;4 cycle delay
rjmp    Enable1c8


Enable1c7:

nop                                             ;1 cycle delay


Enable1c8:                                      ;EXIT this routine (9 total clock cycles for this routine)


;-------------------------

Enable2:                                ;Enable divide by 8

sbrs    r18,1                   ;skip if r18,1(2) is set
rjmp    Enable2c3               ; out -- no action

sbis    PIND,3                  ;skip if PIND2(4) is set - divide by 8
rjmp    Enable2b

sbi             PortB,1                 ;set PortB,1(2)
rjmp    Enable2c7


Enable2b:

cbi             PortB,1                 ;clear PortB,1
rjmp    Enable2c8


Enable2c3:

push    r0
pop             r0                              ;4 cycle delay
rjmp    Enable2c8


Enable2c7:

nop                                             ;1 cycle delay


Enable2c8:                                      ;EXIT this routine


;------------------------------

Enable3:                                ;Enable divide by 9

sbrs    r18,2                   ;skip if r18,2(41) is set
rjmp    Enable3c3               ; out -- no action

sbis    PIND,4                  ;skip if PIND2(4) is set - divide by 7
rjmp    Enable3b

sbi             PortB,1                 ;set PortB,1(2)
```

```
rjmp    Enable3c7

Enable3b:

cbi             PortB,1                 ;clear PortB,1
rjmp    Enable3c8

Enable3c3:

push    r0
pop             r0
rjmp    Enable3c8               ;4 cycle delay

Enable3c7:

nop                                     ;1 cycle delay

Enable3c8:                              ;EXIT this routine

;------------------------------

Enable4:                        ;Enable divide by 10

sbrs    r18,3                   ;skip if r18,3(8) is set
rjmp    Enable4c3               ; out -- no action

sbis    PIND,5                  ;skip if PIND2(4) is set - divide by 7
rjmp    Enable4b

sbi             PortB,1                 ;set PortB,1(2)
rjmp    Enable4c7

Enable4b:

cbi             PortB,1                 ;clear PortB,1
rjmp    Enable4c8

Enable4c3:

push    r0
pop             r0                      ;4 cycle delay
rjmp    Enable4c8

Enable4c7:

nop                                     ;1 cycle delay

Enable4c8:                              ;EXIT this enabling routine

                                        ;38 total clock cycles to here excluding delay chain

;-----------------------------------

sbrs    r18,4                   ;r18,4(16) variable beat frequency flag
rjmp    Vexit41                 ;continue if not set

sbis    PortB,7                 ;Toggle clock input to 4013 IC
rjmp    VB1

cbi             PortB,7                 ;clear PortB,7,(128)
rjmp    VB2

VB1:

sbi             PortB,7                 ;set PortB,7
nop
```

```
VB2:                                        ;END of toggle  46 clock cycles to here

sbrs    r18,5                               ;skip if r18,5(32), search flag is set
rjmp    Vexit49

sbiw    r28,1                               ;decrement Y reg, r28,29 by one
cpi             r29,4                       ;point to turn on check for RC network discharge
brsh    Vexit53

sbic    PIND,1                              ;skip if PIND,1(2) is clear - RC network discharged
rjmp    VB3

cbr             r18,32                      ;clear r18,5(32) - search flag off
rjmp    Vexit57

VB3:

adiw    r28,1
sbiw    r28,1                               ;to check for zero
breq    VB4

rjmp    Vexit62

VB4:

sbiw    r30,1                               ;decrement Z counter, r30,31 (pointer to delay ladder)
cpi             r31,0
brne    Vexit66
cpi             r30,16
brlo    VB6

rjmp    Vexit69

VB6:

ldi             r31,1
ldi             r30,50                      ;reset freq ladder counter to highest frequency

rjmp    Vexit72


; Delay Ladder for VEXITS

Vexit41:Push    r0
                Pop             r0
                Push    r0
                Pop             r0
Vexit49:Push    r0
                Pop             r0
Vexit53:Push    r0
                Pop             r0
Vexit57:Push    r0
                Pop             r0
                nop
Vexit62:Push    r0
                Pop             r0
Vexit66:nop
                nop
                nop
Vexit69:nop
                nop
                nop

Vexit72:                                    ;End of Vexit delay ladder

;End of data section

ijmp                                        ;loop back to correct selected frequency in Z register, r30,31
```

```
;----------------------------------

Initialize1:

;Lowest beat frequency : r31=0  r30 =16
;Highest beat frequency : r31 = 1;  r30 =50

clr             r28
clr             r29                             ;clear Y counter, r28,29
ldi             r31,1
ldi             r30,50                  ;set to highest beat frequency
sbi             PortB,2                 ;OFF variable beat frequency (4013 reset high)

ldi             r16,10
rcall   BlinkLED        ;flash LED 10 times at bootup.

ret


;-----------------------------------

ServiceSwitch:

rcall   ButtonPush
rcall   Decode

mov             r16,r26                 ;get entry number
rcall   BlinkLED

ijmp                                    ;jump back into variable beat frequency loop


;----------------------------

StallButtonPush:        ;stalls and waits for button push entry.

sbic    PIND,0          ;skip if PIND,0, touch plate is touched
rjmp    StallButtonPush

rjmp    ButtonPush


;------------------------------------------------------------

ButtonPush:                     ;Takes in dit, dah, holddown, and coded input

push    r0                      ;number of code digits
push    r1                      ;low code byte
push    r2                      ;next high code byte
push    r3                      ;next high code byte
push    r4                      ;highest code byte
push    r20                     ;holddown count
push    r21                     ;dit count
push    r23                     ;universal counter
push    r24                     ;combined count
push    r28                     ;dah count
push    r30                     ;has total count of entry in Z register, r30,31
push    r31

;r30,r31 (Z register pair contain total value of entry)

clr             r21             ;Dit Count
clr             r20             ;HoldDown count
clr             r23             ;universal counter
clr             r28             ;dah count
clr             r0              ;clear number of bits in code
clr             r1              ;clear lowest code count
clr             r2              ;clear next code count
clr             r3              ;clear next code count
clr             r4              ;clear next code count (clear all code counts)


                                ;r30,31 has total count

B1:

inc             r23                     ;universal counter
```

```
rcall   Delay40
cpi             r23,60          ;delay before first holddown count - 40=one second
brsh    B3                      ;branch if - it is a holddown count

sbis    PIND,0          ;skip if switch is NOT down
rjmp    B1


B2:

cpi             r23,20          ;compare r23 for 20 (1/2 second dah count)
brlo    B2b                     ;branch if lower (dit count)


B2a:                            ;it is dah count

inc             r28                     ;increment dah count

sec                             ;set carry flag -- it is dah entry
rol             r1
rol             r2                      ;rotate data into registers r1-r4
rol             r3
rol             r4

sbi             PortB,0         ;on GREEN LED
rcall   Delay40
rcall   Delay40         ;delay for longer on of LEDs
rcall   Delay40
rcall   Delay40
cbi             PortB,0         ;off LED
clr             r23
inc             r0                      ;increment number of code digits


B2d:                            ;no touch/switch up detection

sbis    PIND,0          ;skip if switch is up
rjmp    B2e                     ;jump back - there is a new entry coming in

inc             r23
rcall   Delay40
cpi             r23,70          ;has switch been up 70 counts? 40=one second (1.75 seconds)
brsh    B4Decode1
rjmp    B2d


BPEnd1:

rjmp    BPEnd


B2e:

clr             r23
rjmp    B1


B2b:                            ;Dit count detected

inc             r21                     ;increment dit count

clc                             ;clear carry flag -- it is dit entry
rol             r1
rol             r2                      ;rotate data into registers r1-r4
rol             r3
rol             r4

sbi             PortB,0
rcall   Delay40
cbi             PortB,0

inc             r0                      ;increment number of code digits (r0)
clr             r23                     ;clear r23
rjmp    B2d


B3:                             ;Holddown count has been detected
```

```
inc             r20                     ;increment holddown count
sbi             PortB,0
rcall   Delay40
rcall   Delay40
cbi             PortB,0         ;flash RED LED

clr             r23                     ;reset r23 to zero count


B4:

inc             r23
rcall   Delay40
sbic    PIND,0          ;skip if switch is down
rjmp    B4a                     ;jump out and look for next entry

cpi             r23,30          ;pause between continuous hold down of switch
brlo    B4

rjmp    B3


B4a:

clr             r23
rjmp    B2d                     ;Check for exit or new entry



B4Decode1:

;This section finalizes.  It combines counts

clr             r30
clr             r31                     ;ready registers for input

mov             r16,r28         ;move dah count in r28 into r16


B4comb1:

tst             r16
breq    B4comb2

adiw    r30,10          ;add value of 10 in r30 to r314
dec             r16
rjmp    B4comb1


B4comb2:

add             r30,r21         ;add value in r21 (dit count) to r24
brcs    B4comb2a
rjmp    Bholdown1


B4comb2a:

inc             r31                     ;carry was set so increment higher byte


Bholdown1:

;Combined value now in Z register pair, r30,31

mov             r16,r20         ;put hold down count into r16

Bholdown1b:

tst             r16                     ;test r20(r16), hold down count
breq    BPEnd
adiw    r30,50          ;add fifty count to total
dec             r16
rjmp    Bholdown1b

;Complete combined value now in Z register pair
```

```
BPEnd:

;Store away in SRAM $00E7-00EB, Dit,Dah,HolddownCount,LSD Total, MSD Total

sts             $0060,r21       ;number of dits
sts             $0061,r28       ;number of dahs
sts             $0062,r20       ;number of hold down counts
sts             $0063,r30       ;LSD of total count
sts             $0064,r31       ;MSD of total count

mov             r26,r30
mov             r27,r31                 ;put low and high bytes of total count into r26 and r27

pop             r31
pop             r30
pop             r28
pop             r24
pop             r23
pop             r21
pop             r20
pop             r4
pop             r3
pop             r2
pop             r1
pop             r0

ret

;  r26 has LSD of total count
;  r27 has MSD of total count


;--------------------------------

Decode:                         ;r26 has entry from push button

; 1 = reset all
; 2 = enable 7th division
; 3 = enable 8th division
; 4 = enable 9th division
; 5 = enable 10th division
; 6 = enable search for variable beat frequency

; 12 = Toggle 7th division on/off
; 13 = Toggle 8th division on/off
; 14 = Toggle 9th division on/off
; 15 = Toggle 10th division on/off
; 16 = Toggle variable beat frequency on/off

Decode1:                        ;reset all -- OFF all frequencies

cpi             r26,1           ;Reset All
brne    Decode2

clr             r18                     ;OFF all frequency divisions

sbi             PortB,7         ;OFF variable beat frequency

ret


;----

Decode2:                        ;Set 7th division flag

cpi             r26,2           ;enable 7th division
brne    Decode3

clr             r18                     ;clear all flags -- off any other selections
sbi             PortB,2         ;off variable frequency
sbr             r18,1           ;on 7th division flag, r18,0(1)

ret

;----

Decode3:                        ;Set 8th division flag
```

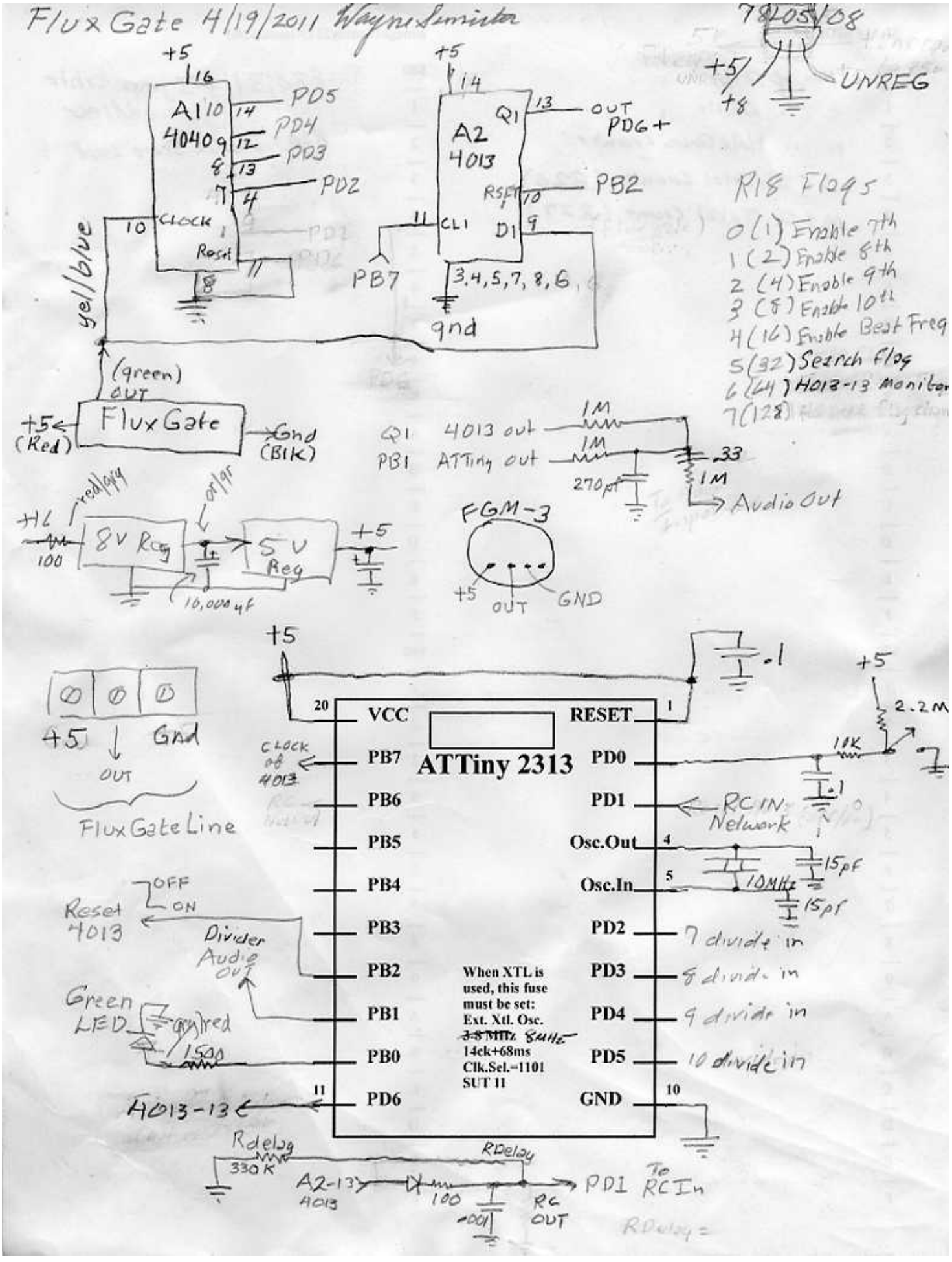
```
cpi             r26,3
brne    Decode4

clr             r18                     ;clear all flags -- off any other selections
sbi             PortB,2         ;off variable frequency
sbr             r18,2           ;on 8th division flag, r18,1(2)

ret

;----

Decode4:                        ;Set 9th division flag

cpi             r26,4
brne    Decode5

clr             r18                     ;clear all flags -- off any other selections
sbi             PortB,2         ;off  variable frequency
sbr             r18,4           ;on variable beat frequency, r18,2(4)

ret

;----

Decode5:                        ;Set 10th division flag

cpi             r26,5
brne    Decode6

clr             r18                     ;clear all flags -- off any other selections
sbi             PortB,2         ;off variable frequency
sbr             r18,8           ;on variable beat frequency, r18,3(8)

ret

;----

Decode6:                        ;enable variable beat frequency

cpi             r26,6
brne    Decode7

clr             r18                     ;clear all flags -- off any other selections

clr             r28
clr             r29                     ;clear repeat same count counter

ldi             r30,50
ldi             r31,1           ;set up highest frequency for start of search

cbi             PortB,2         ;on variable frequency (4013 mixer)

sbr             r18,16          ;on variable beat frequency, r18,4(16)
sbr             r18,32          ;on search flag r18,5(32)

ret


;----

Decode7:                        ;FETCH previously saved EEPROM data for Beat Frequency

cpi             r26,7
brne    Decode8

ldi             r16,$00         ;address for EEPROM data (r30)
rcall   FetchEEPROM
mov             r30,r16         ;move Fetched Date into r30

ldi             r16,$01         ;addess for EEPROM data (r31)
rcall   FetchEEPROM
mov             r31,r16         ;move fetched data into r31

clr             r18                     ;clear all flags -- off any other selections

cbi             PortB,2         ;on variable frequency (4013 mixer)

sbr             r18,16          ;on variable beat frequency, r18,4(16)
```

```
cbr             r18,32          ;OFF search flag r18,5(32)

ret


;------

Decode8:                        ;RECORD current Beat Frequency Data to EEPROM

cpi             r26,8
brne    Decode12

mov             r16,r30         ;get beat freq. data from r30 into r16
ldi             r17,$00         ;address for r30 data storage in EEPROM
rcall   WriteEEPROM

mov             r16,r31         ;get beat freq. data from r31 into r16
ldi             r17,$01         ;address for r31 data stgorage in EEPROM
rcall   WriteEEPROM

ldi             r16,2           ;for two blinks of LED
rcall   BlinkLED

ret


;-------

Decode12:                       ;7th division.  Toggle of bit (add to present)

cpi             r26,12
brne    Decode13

sbrs    r18,0           ;skip if r18,0(1) is set
rjmp    Decode12a

cbr             r18,1           ;clear r18,0(1)

ret


Decode12a:

sbr             r18,1           ;set r18,0(1)

ret


;----

Decode13:                       ;8th division.  Toggle of bit (add to present)

cpi             r26,13
brne    Decode14

sbrs    r18,1           ;skip if r18,1(2) is set
rjmp    Decode13a

cbr             r18,2           ;clear r18,1(2)

ret


Decode13a:

sbr             r18,2           ;set r18,1(2)

ret


;----

Decode14:                       ;9th division.  Toggle of bit (add to present)

cpi             r26,14
brne    Decode15

sbrs    r18,2           ;skip if r18,2(4)
```

```
rjmp    Decode14a

cbr             r18,4           ;clear r18,2(4)

ret


Decode14a:

sbr             r18,4           ;set r18,2(4)

ret


;----

Decode15:               ;8th division.  Toggle of bit (add to present)

cpi             r26,15
brne    Decode16

sbrs    r18,3           ;skip if r18,3(8) is set
rjmp    Decode15a

cbr             r18,8           ;clear r18,3(8)

ret


Decode15a:

sbr             r18,8           ;set r18,3(8)

ret


;----

Decode16:               ;Variable beat.  Toggle of bit (add to present)

cpi             r26,16
brne    DecodeEnd

sbrs    r18,4           ;skip if r18,4(16) is set
rjmp    Decode16a

cbr             r18,16          ;clear r18,4(16) - Beat Freq. disable
sbi             PortB,2         ;set PortB,2 for variable beat off

ret


Decode16a:

sbr             r18,16          ;set r18,4(16) - Beat Freq. enable
sbr             r18,32          ;on search flag r18,5(32)

ldi             r30,50
ldi             r31,1           ;set up highest frequency for start of search

cbi             PortB,2         ;on PortB,2 - 4013 enable

clr             r28
clr             r29                     ;clear Y register, repeat count counter

ret


;----

DecodeEnd:

ldi             r16,10
rcall   BlinkLED ;TEN blinks for non defined entry

ret
```

```
;-----------------------------------------------

;Delay subroutines

;-------------------------

Delay40:                                ;For 10MHz crystal frequency

push    r30
push    r31

ldi             r30,31
ldi             r31,244


D40a:

sbiw    r30,1
brne    D40a

pop             r31
pop             r30

ret


;-------------------------------------------

DelaySixth:                                     ;<1/6 th second delay

push    r16

ldi             r16,6


DHa:

rcall   Delay40
dec             r16
brne    DHa

pop             r16

ret


;-------------------------------------------

DelayOne:                       ;ONE second delay

push    r16
ldi             r16,40

rjmp    DHa

 ;-----------------------------------------

 DelayTwo:                              ;FIVE second delay

 push    r16
 ldi     r16,80
 rjmp    DHa


 ;-----------------------------------------

FetchEEPROM:            ;grabs data in permanent saved EEPROM
                                        ;r16 has address location - Then r16 returns data

cli                                     ;clear interrupts

FEE:

sbic    EECR,EEWE       ;Wait for completion of previous write
rjmp    FEE

out             EEAR,r16 ;set up address in address register
sbi             EECR,EERE       ;start EEPROM read by writing EERE
```

```
in              r16,EEDR ;read data from data register

sei                                     ;on interrupts

ret                                     ;code data now in r16


;---------------------------------------------------

WriteEEPROM:            ;write to EEPROM - r17 has memory location, r16 has data

cli                                     ;off interrupts


WEE:

sbic    EECR,EEPE       ;Wait until ready for write
rjmp    WEE

out             EEAR,r17 ;set up address in address register
out             EEDR,r16        ;put contents of r16 into EEPROM data location

sbi             EECR,EEMPE      ;write logical one to EEMPE

sbi             EECR,EEPE       ;start eeprom write by setting EEPE

sei                                     ;on interrupts

ret


;---------------------------------------------------

BlinkLED:               ;Blinks GREEN LED number of times in r16

sbi             PortB,0         ;ON green LED
rcall   Delay40
rcall   Delay40
cbi             PortB,0         ;OFF green LED
rcall   DelaySixth
dec     r16
brne    BlinkLED

ret


;-----------------------------------------------------------
```


FluxGate 4/19/2011 Wayne Limiter
7805/08
+5
UNREG
+8
A1 4040
A2 4013
PD5
PD4
PD3
PD2
PB2
PB7
3,4,5,7,8,6
gnd
yel/blue
(green) OUT
FluxGate
+5 (Red)
Gnd (Blk)
R18 Flags
0 (1) Enable 7th
1 (2) Enable 8th
2 (4) Enable 9th
3 (8) Enable 10th
4 (16) Enable Beat Freq
5 (32) Search Flag
6 (64) 4013-13 Monitor
7 (128)
Q1 4013 out
PB1 ATTiny out
1M
.33
270pf
Audio Out
8V Reg
5V Reg
10,000 uf
FGM-3
OUT
GND
FluxGate Line
ATTiny 2313
VCC
RESET
PB7
PB6
PB5
PB4
PB3
PB2
PB1
PB0
PD6
PD0
PD1
Osc.Out
Osc.In
PD2
PD3
PD4
PD5
GND
When XTL is used, this fuse must be set:
Ext. Xtl. Osc.
8MHz
14ck+68ms
Clk.Sel.=1101
SUT 11
Reset 4013
OFF
ON
Divider Audio Out
Green LED
1500
4013-13
RC IN Network
10MHz
15pf
7 divide in
8 divide in
9 divide in
10 divide in
Rdelay
330K
A2-13 4013
100
.001
RC OUT
To RC In
RDelay=