

```
; Disassembled by:
; DASMx object code disassembler
; (c) Copyright 1996-1999 Conquest Consultants
; Version 1.30 (Oct 6 1999)
;
; File: C:\Users\Zack\Downloads\Defender Sound Research\defenderb ROM\defend -
Copy.snd
;
; Size: 2048 bytes
; Checksum: BB43
; CRC-32: 6479F442
;
; Date: Sun Sep 08 21:11:37 2019
;
; CPU: Motorola 6808 (6800/6802/6808 family)
ORG $0000
X0000
ORG $0001
X0001
ORG $0002
X0002
ORG $0003
X0003
ORG $0004
flg04
ORG $0005
X0005
ORG $0006
X0006
ORG $0007
flg07
ORG $0008
X0008
ORG $0009
ALFSR
ORG $000A
BLFSR
ORG $000B ;pointer hi
pnt0B
ORG $000C ;lo
p0Blo
ORG $000D ;pointer hi
pnt0D
ORG $000E ;lo
p0Dlo
ORG $000F ;pointer hi
pnt0F
ORG $0010 ;lo
p0Flo
ORG $0011
X0011
ORG $0012
X0012
ORG $0013 ;$13 - $1B
buf13
X0013
ORG $0014
X0014
ORG $0015
X0015
ORG $0016 ;pointer hi
pnt16
X0016
ORG $0017 ;lo
p16lo
X0017
ORG $0018 ;pointer hi
pnt18
ORG $0019 ;lo
p18lo
ORG $001A
X001A
ORG $001B ;pointer hi
pnt1B
ORG $001C ;lo
```

```

p1Blo
X001C
    ORG $001D    ;pointer hi
pnt1D
X001D
    ORG $001E    ;lo
p1Dlo
    ORG $001F    ;pointer hi
pnt1F
    ORG $0020    ;lo
p1Flo
    ORG $0021
cnt21
    ORG $0022
cnt22
    ORG $0023
X0023
    ORG $0024
buf24

    ORG $0400
PIA_BASE

    ORG $0402
PIA_PORB

    ORG $F800

    nop
RST:
    sei
    lds    #$007F    ;set stack pointer
    ldx    #PIA_BASE
    clr    $01,x    ;set PIA CRA reg, clear inrq, select DDRA
    clr    $03,x    ;set PIA CRB reg, clear inrq, select DDRB
    ldaa   #$FF
    staa   $00,x    ;set PORTA to outputs
    clr    $02,x    ;set PORTB to outputs
    ldaa   #%00110111 ;Set CRB, CB2 goes low as write CR reg-
    staa   $03,x    ;select PORTB, low-to-hi IRQ, enable IRQ
    ldaa   #%00111100 ;Set CRA, CA2 goes hi as write CR reg-
    staa   $01,x    ;select PORTA, disable IRQ
    staa   ALFSR    ;$09 LFSR init?
    clra
    staa   flg07    ;clear x0007
    staa   flg04    ;clear x0004
    staa   X0005    ;clear
    staa   X0006    ;clear
    staa   X0008    ;clear
    cli
LOOP:
    bra LOOP
;-----

;A holds value for STEP_PNT0D
;will multiply by 9
;subroutine
;load buf13 from table0
Lod13:
LF82A:
    tab    ;B = A
    asla
    asla
    asla    ;A *= 8
    aba    ;A = 9A
    ldx    #buf13
    stx    pnt0F    ;init to buf13
    ldx    #TABLE0 ;LFB0A loads pnt0D from here
    jsr    STEP_PNT0D ;add 9A to pnt0D
    ldab   #$09    ;copy 9 bytes from #TABLE0 to buf13
    jmp    Cpy0F    ;ENDs LF82A Subroutine
;-----

;Generate Square wave (Extra life?)
;$13, $14 hold hi and lo time delays

```

```

;$15 (#0?)
;$16?, $17?
;pnt18 hold sync freq
;$1A
;$1B holds amplitude
;subroutine
LF83F:
    ldaa    pnt1B        ;amplitude
    staa    PIA_BASE     ;store $1B val
LF844:
    ;V delay loop init p1
    ldaa    X0013        ;hi time delay, move to $1C
    staa    X001C        ;store $13 val
    ldaa    X0014        ;lo time delay, move to $1D
    staa    X001D        ;store $14 val
LF84C:
    ldx     pnt18
LF84E:
    ;V delay loop init p2
    ldaa    X001C
    com     PIA_BASE     ;invert
LF853:
    ;V 'hi' delay
    dex     ;pnt18 val
    beq     LF866        ;sync check
    deca    ;X001C ($13) val
    bne     LF853        ;^
    com     PIA_BASE     ;invert
    ldaa    X001D
LF85E:
    ;V 'lo' delay
    dex     ;pnt18 val
    beq     LF866        ;sync check
    deca    ;X001D ($14) val
    bne     LF85E        ;^
    bra     LF84E        ;^^
;
LF866:
    ldaa    PIA_BASE
    bmi     LF86C        ;if hi, sync square
    coma    ;invert PIA value
LF86C:
    ;^
    adda    #$00        ;is carry ever set??
    staa    PIA_BASE     ;store (inverted?) PIA (+1?)
    ldaa    X001C        ;hi delay val
    adda    X0015        ;add to $1C val (always 0)
    staa    X001C        ;store $15 val + $1C val

    ldaa    X001D        ;lo delay val
    adda    X0016        ;add $1D val
    staa    X001D        ;store $1D val + $16

    cmpa    X0017
    bne     LF84C
    ldaa    X001A
    beq     LF88B_rts
    adda    X0013        ;add val from $1A
    staa    X0013        ;store $13 val + $1A val
    bne     LF844
LF88B_rts:
    rts
;END LF83F Subroutine
;-----

;White noise routines?
;Indirect Subroutine offset 3
LF88C:
    ldaa    #$01
    staa    X001A        ;set to #1
    ldab    #$03
    bra     LF89E
;
;Indirect Subroutine offset 7
LF894:

```

```

    ldaa    #$FE
    staa    X001A    ;set to #$FE (#-2)
    ldaa    #$C0
    ldab    #$10
    bra     LF89E
;
LF89E:
    ;only called from directly above^
    staa    p18lo    ;a = 1 or -2
    ldaa    #$FF
    staa    PIA_BASE    ;set to max #$FF
    stab    X0015    ;init to #3 or #16
LF8A7:
    ldab    X0015    ;LFSR counter
LF8A9:
    ;Linear Feedback Shift Register??
    ldaa    BLFSR
    lsra
    lsra
    lsra
    eora    BLFSR
    lsra
    ror     ALFSR
    ror     BLFSR
    bcc     LF8BC
    com     PIA_BASE    ;invert
LF8BC:
    ldaa    p18lo
LF8BE:
    deca    ;delay?
    bne     LF8BE
    decb    ;count down from #3 or #16
    bne     LF8A9
    ldaa    p18lo
    adda    X001A    ;add val from $19
    staa    p18lo    ;add val from $1A
    bne     LF8A7
    rts
;END LF88C and LF894 subroutines
;-----

;Indirect Subroutine offset 6
LF8CD:
    ldaa    #$20
    staa    X0015    ;set to #32
    staa    pnt18    ;set to #32
    ldaa    #$01
    ldx     #$0001
    ldab    #$FF
    bra     LF8DC ; jump to very next address?
;
LF8DC:
    ;only called from directly above^
    staa    X0013    ;init to #1
LF8DE:
    ;local
    stx     pnt16    ;init to #1
LF8E0:
    ;local
    stab    X0014    ;init to #$FF, or store $14-$13. gets written to PIA_BASE
    ldab    X0015    ;LFSR counter
LF8E4:
    ;Linear Feedback Shift Register??
    ldaa    BLFSR
    lsra
    lsra
    lsra
    eora    BLFSR
    lsra
    ror     ALFSR
    ror     BLFSR
    ldaa    #$00
    bcc     LF8F8
    ldaa    X0014    ;load to store in PIA_BASE
LF8F8:

```

```

;local
staa    PIA_BASE    ;store #0 or $14 val (noise?)
ldx     pnt16
LF8FD:
;loop
dex     ;countd $16 val down to 0, for delay?
bne     LF8FD      ;loop
dec     ;$15 or $14 - $13
bne     LF8E4
ldab    X0014
subb    X0013      ;sub #1 from $14 ($13 is always 1 here)
beq     LF912
ldx     pnt16
inx
ldaa    pnt18      ;should equal #32 here, maybe IRQ can change it?
beq     LF8E0
bra     LF8DE
LF912:
rts
;END LF8CD Subroutine
;-----

;Indirect Subroutine offset 1
;jmpd to from LFD12, also in branch table
LF913:
ldab    #$01      ;to store in $13
stab    flg04      ;store #1
clra
staa    p18lo      ;clear, set to #0
bra     LF930
;
;Indirect Subroutine offset 8
LF91C:
clra
staa    p18lo      ;clear, set to #0
ldab    #$03      ;to store in $13
bra     LF930
;
;Indirect Subroutine offset 9
LF923:
ldaa    #$01
staa    p18lo      ;set to #1
ldx     #$03E8      ;Something external. Speech?
ldaa    #$01
ldab    #$FF      ;to store in $13
bra     LF930
;
LF930:
;only jumpd to from above^
staa    pnt18      ;LF913 or LF91C a=0, else LF923 a=1
stab    X0013      ;init to #1, #3, or #255
stx     pnt16      ;#$03E8 if called by LF923. Else #$F913 or #$F91C
clr     X0015
LF939:
ldx     pnt16
ldaa    PIA_BASE
LF93E:
;Linear Feedback Shift Register??
tab
lsrb
lsrb
lsrb
eorb    BLFSR
lsrb
ror     ALFSR
ror     BLFSR
ldab    X0013
tst     p18lo
beq     LF954
andb    ALFSR      ;and with $13 value
LF954:
;local
stab    X0014      ;gets written to PIA_BASE
ldab    X0015      ;load to add to $14 (?)
cmpa    BLFSR

```

```

    bhi     LF96E
LF95C:
    ;local
    dex
    beq     LF985
    staa    PIA_BASE    ;store $0A val (Noise?)
    addb    X0015    ;add to $15 and $14
    adca    X0014    ;add to $15
    bcs     LF97E
    cmpa    BLFSR
    bls     LF95C
    bra     LF97E
;
LF96E:
    dex
    beq     LF985
    staa    PIA_BASE
    subb    X0015    ;sub from $15 based val
    sbca    X0014    ;sub from $15
    bcs     LF97E
    cmpa    BLFSR
    bhi     LF96E
LF97E:
    ldaa    BLFSR    ;gets written to PIA_BASE
    staa    PIA_BASE    ;store $0A val (Noise?)
    bra     LF93E
;
LF985:
    ldab    pnt18
    beq     LF93E
    ldaa    X0013    ;load to combine with $15 val
    ldab    X0015    ;combine with $13
    lsra    ;/2
    rorb    ;shift 3 LSB of $13 val into B MSB
    lsra    ;/4
    rorb
    lsra    ;/8
    rorb
    coma    ;2's comp
    negb
    sbca    #$FF
    addb    X0015
    adca    X0013
    stab    X0015
    staa    X0013
    bne     LF939    ;write noise to PIA_BASE
    cmpb    #$07
    bne     LF939    ;write noise to PIA_BASE
    rts     ;returns from LFA48
; END LF913, LF91C, LF923 and LFA48 Subroutines
;-----

;Indirect Subroutine offset 10
;write #$FD9A-#$FDA9 to PIA_BASE
LF9A6:
    ldaa    #$FD
    staa    pnt0F    ;init to #$FDXX
    ldx     #$0064    ;#100
    stx     pnt0B    ;init to #100 #$64
LF9AF:
    ;local
    addb    p0B10
    ldaa    X0011
    adca    pnt0B    ; 256 > $0B > 100
    staa    X0011    ;store $11 val + $0B val + carry from $0C
    ldx     pnt0B
    bcs     LF9BF
    bra     LF9BD    ;jump to very next addr?
LF9BD:
    ;local
    bra     LF9C2
;
LF9BF:
    ;local
    inx     ;count up from 100

```

```

    beq     LF9D3
LF9C2:
    ;local
    stx     pnt0B    ;store $0B (+1)
    anda    #$0F     ;$11 val & %1111
    adda    #$9A
    staa    p0Flo    ;set pnt05 to #$FD9A-#$FDA9
    ldx     pnt0F
    ldaa    $00,x
    staa    PIA_BASE    ;store value from #$FD9A-#$FDA9
    bra     LF9AF
LF9D3:
    rts
;END F9A6 Subroutine
;-----

;Indirect Subroutine offset 11
;Generates squarewave?
LF9D4:
    clra
    staa    PIA_BASE    ;store 0
    staa    X0011        ;store 0, min delay time/freq for tone
LF9DA:
    clra
LF9DB:
    cmpa    X0011
    bne     LF9E2
    com     PIA_BASE    ;invert value
LF9E2:
    ldab    #$12
LF9E4:
    ;delay loop?
    decb
    bne     LF9E4
    inca
    bpl     LF9DB
    com     PIA_BASE    ;invert value
    inc     X0011        ;decrease pitch
    bpl     LF9DA
    rts
;END F9D4 Subroutine
;-----

;Indirect Subroutine offset 12
LF9F3:
    ldx     #buf13
LF9F6:
    clr     $00,x    ;clear $13-$1B
    inx
    cpx     #$001B
    bne     LF9F6
    ldaa    #$40
    staa    X0013    ;store #64
LFA02:
    ldx     #buf13
    ldaa    #$80
    staa    X0011    ;store #128
    clrb
LFA0A:
    ldaa    $01,x    ;starts at $14
    adda    $00,x    ;starts at $13
    staa    $01,x    ;starts at $14
    bpl     LFA14
    addb    X0011    ;starts at #128
LFA14:
    lsr     X0011    ;divide by 2
    inx
    inx
    cpx     #$001B    ;loop from $13-$1B
    bne     LFA0A
    stab    PIA_BASE    ;store $11 val + /2 $11 val + /2 $11...
    inc     X0012        ;gets written to PIA elsewhere
    bne     LFA02
    ldx     #buf13
    clrb

```

```

LFA2A:
    ;local
    ldaa    $00,x
    beq     LFA39 ;v
    cmpa    #$37
    bne     LFA36 ;v
    ldab    #$41
    stab    $02,x    ;store #$41 at $13-$1B + 2
LFA36:
    dec    $00,x
    incb
LFA39:
    inx
    inx
    cpx    #$001B
    bne     LFA2A ;^
    tstb
    bne     LFA02 ;reset x to #$13
    rts
;END F9F3 Subroutine
;-----

;Indirect Subroutine offset 13
LFA44:
    dec     X0008
    rts
; -----

;only called from IRQ if $08 val < 0
;A holds (triggers)loop count #$1E, stored in $11
;subroutine, rts near LF985?
LFA48:
    clr     X0008
    staa    X0011    ;store trigger values
    ldx     #TABLE1 ;$FDAA start of a table
LFA50:
    ldaa    $00,x    ;load from TABLE1
    beq     LFA81    ;table zero terminated, exit if end
    dec     X0011
    beq     LFA5F    ;if $11 = 0
    inca
    jsr     STEP_PNT0D ;add A to pnt0d
    bra     LFA50
;
LFA5F:
    inx
    stx     pnt0F    ;set to #TABLE1($FDAA) + 1
    jsr     STEP_PNT0D ;add A to pnt0d
    stx     pnt0D    ;set to #TABLE1($FDAA) + 1 + A(?)
    ldx     pnt0F    ;load #TABLE1($FDAA) + 1
LFA69:
    ldaa    $00,x    ;val for $15
    staa    X0015
    ldaa    $01,x    ;loop count for LFAB3
    ldx     $02,x    ;val for $13
    stx     X0013    ;set to val from TABLE1?
    bsr     LFAB3
    ldx     pnt0F
    inx
    inx
    inx
    inx
    stx     pnt0F    ;add 4 to x and pnt0F
    cpx     pnt0D
    bne     LFA69
LFA81:
    jmp     LFD0E
;-----

;Indirect Subroutine offset 14
LFA84:
    ldaa    #$03
    staa    X0008    ;set to #3
    rts
;END FA84 Subroutine

```



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

```
;only called from IRQ
```

```
;A = #$1D
```

```
;subroutine
```

```
LFA89:
```

```
    dec    X0008
    beq    LFA9A
    ldab   X0015
    aslb   ;*2
    aslb   ;*4
    aslb   ;*8
    aslb   ;*16
    aba    ;A = #$1D + $15 val * 16
    staa   X0015    ; store #$1D + $15 val * 16
    clra
```

```
LFA98:
```

```
    ;Loop
    bra    LFA98    ;loop
```

```
;
```

```
LFA9A:
```

```
    ;$08 = #0
    deca
    cmpa   #$0B    ;TABLE2 length
    bls    LFAA0
    clra
```

```
LFAA0:
```

```
    ldx    #TABLE2    ;$FE41
    jsr    STEP_PNT0D
    ldaa   $00,x    ;load from TABLE2 + A(?)
    ldx    #$FFFF
    stx    X0013    ;set to #$FFFF for LFADB
    bsr    LFAB3    ;A holds rate
```

```
LFAAF:
```

```
    ;loop
    bsr    LFADB
    bra    LFAAF    ;loop
```

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

```
;A = delay time
```

```
;subroutine, bsr to here
```

```
;generates delay routine in ram
```

```
LFAB3:
```

```
    ldx    #$0016    ;delay routine starts here
```

```
LFAB6:
```

```
    ;v
    cmpa   #$00
    beq    LFACF    ;v
    cmpa   #$03
    beq    LFAC7    ;v
    ldab   #$01    ;write NOPs to RAM to generate dynamic delay
    stab   $00,x    ;store NOP at x init to #$16
    inx
    suba   #$02
    bra    LFAB6    ;^
```

```
;
```

```
LFAC7:
```

```
    ;^
    ldab   #$91
    stab   $00,x    ;write CMPA $0000 to RAM x=(A-3)/2 + #$16
    clr    $01,x
    inx
    inx
```

```
LFACF:
```

```
    ;^
    ldab   #$7E    ;write JMP $FADD to RAM ($16 if a=0) ($18 if a=3)
    stab   $00,x    ;store #$7E
    ldab   #$FA
    stab   $01,x    ;store #$FA
    ldab   #$DD
    stab   $02,x    ;store #$DD
```

```
;$12 = PIA start value
```

```
;$13 = loop count ($FFFF)
```

```

LFADB:
;subroutine, bsr to here from LFAAF
ldx    X0013    ;$13 = #$FFFF
;FADD
clra
db $F6,$00,$12
;ldab    $0012
incb    ;$12 + 1
stab    X0012    ;$12 + 1
andb    X0015    ;and with $12
lsrb    ;\2
adca    #$00
lsrb    ;\4
adca    #$00
lsrb    ;\8
adca    #$00
lsrb    ;\16
adca    #$00
lsrb    ;\32
adca    #$00
lsrb    ;\64
adca    #$00
lsrb    ;\128
adca    #$00    ;A = count of 1s in B
aba     ;A = A + B
asla    ;*2
asla    ;*4
asla    ;*8
asla    ;A * 16
staa    PIA BASE    ;store manipulated $12 val
dex     ;$13 val -1
beq     LFB09    ;return
jmp     pnt16    ;jump to dynamic delay routine in RAM
LFB09:
rts
;END LFA89 and LFAB3 Subroutines
;-----

;subroutine. Ends LF82A, called from LFC75 too
Cpy0F:
LFB0A:
psha
LFB0B:
;copy from pnt0d to pnt0f
;B holds number of bytes
;x points to TABLE0, if called from LF82A
ldaa    $00,x    ;x = pnt18 val + 1, if called from LFC75
stx     pnt0D
ldx     pnt0F
staa    $00,x    ;x = pnt0F
inx
stx     pnt0F
ldx     pnt0D
inx
decb
bne     LFB0B    ;^
pula
rts
;END LFB0A and LF82A Subroutines
;-----

;Indirect Subroutine offset 5
LFB1E:
clra
staa    flg04    ;clear
staa    X0005    ;clear
rts
;-----

;Indirect Subroutine offset 2
LFB24:
clr     flg04
ldaa    X0005
anda    #$7F
cmpa    #$1D

```

```

        bne     LFB30
        clra
LFB30:
        inca
        staa    X0005    ;store 1 or val +1
        rts
;END LFB24 Subroutine
;-----

LFB34:
        ldaa    #$0E
        jsr     LFB81    ;Loads a set of values from TABLE4
        ldaa    X0005    ;$05 can = 0-29
        asla
        asla
        coma    ;A = -($05 val * 4)
        jsr     LFC39    ;store A in X0023
LFB41:
        ;Loop
        inc     p16lo
        jsr     LFC3B
        bra     LFB41    ;Loop
;-----

;Indirect Subroutine offset 0
;subroutine that never rts. Loop instead?
LFB49:
        ldaa    #$03
        jsr     L0d13    ;load buf13 from table0 offset 3
        ldab    X0006
        cmpb    #$1F
        bne     LFB55
        clrb
LFB55:
        ;local
        incb
        stab    X0006    ;inc $06, set to #1 if = 31
        ldaa    #$20
        sba     ;A -= B
        clrb
LFB5C:
        ;local
        cmpa    #$14
        bls     LFB65
        addb    #$0E
        deca
        bra     LFB5C
;
LFB65:
        ;local
        addb    #$05
        deca
        stab    X0013    ;param for LF83F sbrrt
        bne     LFB65
LFB6C:
        ;Loop
        jsr     LF83F    ;Write $1B($FF) to PIA_BASE, $13, $14, $15, $16
        bra     LFB6C    ;Loop
;-----

;Indirect Subroutine offset 4
        ldaa    flg07    ;exit routine if != 0
        bne     LFB7E
        inc     flg07    ;set to #1
        ldaa    #$0D    ;will become $5B and work to inc PNT0D
        bsr     LFB81    ;loads values from TABLE4
        bra     LFB7E    ;Writes buf24 sample to PIA_BASE
LFB7E:
        jmp     LFC2E
;
;A holds value for STEP_PNT0D
;loads set of values from TABLE4
;subroutine
LFB81:
        tab     ;B = A

```

```

    aslb      ;double B
    aba      ;A += B
    aba
    aba
    ldx      #TABLE4
    jsr      STEP_PNT0D ;A = A * 7, will add to pointer
    ldaa     $00,x      ;load from TABLE4 +(A*7)
    tab      ;B = A
    anda     #$0F       ;low nyble > $14
    staa     X0014      ;used in LFBF7 count/delay?
    lsr      ;/2
    lsr      ;/4
    lsr      ;/8
    lsr      ;/16
    stab     X0013      ;store hi nyble in $13
    ldaa     $01,x      ;load from TABLE4 +1
    tab      ;B = A
    lsr      ;/2
    lsr      ;/4
    lsr      ;/8
    lsr      ;/16
    stab     X0015      ;$15 gets hi nyble (flag for LFBF7)
    anda     #$0F
    staa     X0011      ;$11 gets lo nyble
    stx      pnt0B      ;#TABLE4 (+1?)
    ldx      #TABLE3    ;$FE4D
LFBAB:
    ;loop, get address of wave to subtract
    dec      X0011      ;max 6
    bmi      LFB8      ;if $11 < 0
    ldaa     $00,x      ;load wav length from TABLE3
    inca
    jsr      STEP_PNT0D ;add length + 1 to addr to get next wave
    bra      LFBAB      ;loop
;
LFB8:
    ;local
    stx      pnt18      ;#TABLE3 wave to subtract
    jsr      LDWAV      ;copy pnt18(TABLE3) # bytes to #$24
    ldx      pnt0B      ;TABLE4 + (?)
    ldaa     $02,x      ;TABLE4 subtraction multiplier
    staa     X001A      ;subtraction multiplier (used for volume?)
    jsr      SubWAV     ;subtract pnt18 wave from buf24 wave
    ldx      pnt0B
    ldaa     $03,x      ;TABLE4 hi pointer?
    staa     pnt16
    ldaa     $04,x      ;TABLE4
    staa     p16lo      ;load from somewhere in TABLE4???
    ldaa     $05,x      ;TABLE4
    tab
    ldaa     $06,x      ;TABLE4, offset for TABLE5
    ldx      #TABLE5    ;$FF55
    jsr      STEP_PNT0D
    tba
    stx      pnt1B      ;TABLE5 + A
    clr      X0023
    jsr      STEP_PNT0D
    stx      pnt1D      ;TABLE5 end
    rts
; END LFB81 Subroutine
;-----

;$13 hold loop counter
;$16 & $17 hold loop countes
;$1B holds pointer
; subroutine
LFBF7:
    ldaa     X0013      ;loop count
    staa     cnt22      ;store $13 for loop count
LFBEB:
    ldx      pnt1B      ;TABLE5 pointer
    stx      pnt0D      ;TABLE5 pointer
LFBF7:
    ;local
    ldx      pnt0D      ;TABLE5

```

```

    ldaa    $00,x    ;load from TABLE5
    adda    X0023    ;add to TABLE5 val (init 0)
    staa    cnt21    ;store delay/speed value
    cpx     pnt1D    ;TABLE5 end
    beq     LFC21
    ldab    X0014    ;0 - 16
    inx     ;inc pnt0D
    stx     pnt0D    ;TABLE5
LFC00:
    ;local
    ldx     #buf24
LFC03:
    ;local
    ldaa    cnt21
LFC05:
    deca    ;delay
    bne     LFC05
    ldaa    $00,x    ;load from buf24
    staa    PIA_BASE
    inx
    cpx     pnt1F    ;end of sample
    bne     LFC03
    decb    ;dec val from $14
    beq     LFBEB    ;local ^
    inx     ;delay
    dex
    inx
    dex
    inx
    dex
    inx
    dex
    inx
    dex
    nop
    nop
    bra     LFC00    ;local ^
;
LFC21:
    ldaa    X0015    ;subtraction multiplier
    bsr     SubWAV    ;subtract pnt18 wave from buf24 wave
    dec     cnt22
    bne     LFBEB
    ldaa    flg07    ;return if $07 != #0
    bne     LFC74    ;return
LFC2E:
    ldaa    pnt16    ;return if $16 = 0
    beq     LFC74    ;return
    dec     p16lo    ;return if $17 = 0
    beq     LFC74    ;return
    adda    X0023    ;add to $16 val
LFC39:
    ;subroutine, only called from LFB34
    staa    X0023    ;store $23+$16 or -($05 *4)
LFC3B:
    ;subroutine
    ldx     pnt1B    ;TABLE5?
    clrb
LFC3E:
    ldaa    X0023
    tst     pnt16
    bmi     LFC4B
    adda    $00,x    ;add to $23 val
    bcs     LFC51
    bra     LFC56
;
LFC4B:
    adda    $00,x
    beq     LFC51
    bcs     LFC56
LFC51:
    tstb
    beq     LFC5C
    bra     LFC65
;
LFC56:
    tstb

```

```

    bne     LFC5C
    stx     pnt1B    ;TABLE5?
    incb
LFC5C:
    inx
    cpx     pnt1D    ;TABLE5 end
    bne     LFC3E
    tstb
    bne     LFC65
    rts      ;END LFB7, LFC39 and LFC3B Subroutines
;
LFC65:
    stx     pnt1D    ;TABLE5 end?
    ldaa    X0015
    beq     LFC71
    bsr     LDWAV    ;loads a wave into buf24
    ldaa    X001A    ;subtraction multiplier
    bsr     SubWAV   ;subtract pnt18 wave from buf24 wave
LFC71:
    jmp     LFB7     ;Writes buf24 sample to PIA_BASE
LFC74:
    rts
;END LFB7, LFC39 and LFC3B Subroutines
;-----

;subroutine start
; copy from pnt18 to $24
LDWAV:
LFC75:
    ldx     #buf24
    stx     pnt0F    ;store #$24, destination for copy
    ldx     pnt18    ;Points to TABLE3($FE4D) if called from LFB8
    ldab    $00,x    ;x = pnt18, load #bytes to copy in LFB0A
    inx
    jsr     Cpy0F    ;copy B bytes from pnt18 to $24
    ldx     pnt0F
    stx     pnt1F    ;store pnt0F val
    rts
;END LFC75 Subroutine
;-----

;A holds val for $12 and $11 subtraction multiplier
;pnt18 points to wave to subtract from buf24
;pnt1F holds end of sample in buf24 array
;subroutine start
; subtract pnt18 wave from buf24
SubWAV:
LFC87:
    tsta
    beq     LFCB5
    ldx     pnt18    ;load to write to pnt0D
    stx     pnt0D    ;store pnt18 val
    ldx     #buf24
    staa    X0012
LFC93:
    stx     pnt0F    ;init to #$24
    ldx     pnt0D    ;counts up from pnt18 val
    ldab    X0012
    stab    X0011    ;store $12 val
    ldab    $01,x    ;x = pnt0D
    lsr     ;/2
    lsr     ;/4
    lsr     ;/8
    lsr     ;/16
    inx
    stx     pnt0D    ;pnt0d + 1
    ldx     pnt0F
    ldaa    $00,x    ;buf24
LFC98:
    ;local
    sba     ;A -= B
    dec     X0011    ;A -= B * $11 val
    bne     LFC98
    staa    $00,x    ;buf24
    inx

```

```

        cpx     pnt1F    ;end of buf24 sample
        bne     LFC93
LFCB5:
        rts
;END LFB E7 and LFC87 Subroutines
;-----

IRQ:
        lds     #$007F    ;set stack pointer
        ldaa    PIA_PORB    ;$0402 Trigger inputs
        cli
        coma     ;invert A
        anda    #$1F      ;0b011111, lowest 5 bits
        ldab    X0008
        beq     LFCCD      ;$08 = 0
        bpl     LFCC9      ;$08 > 0
        jsr     LFA48      ;$08 < 0, clr $08
LFCC9:
        ;$08 > 0
        deca
        jsr     LFA89      ;writes $12(after manip) to PIA_BASE, dec $08
LFCCD:
        ;$08 = 0
        clrb
        cmpa    #$0E      ;0b1110
        beq     LFCD4      ;V
        stab    X0006      ;clear
LFCD4:
        ;^
        cmpa    #$12      ;0b10010
        beq     LFCDA      ;V
        stab    flg07      ;clear
LFCDA:
        ;^
        ldab    $EFFF      ;speech synth ROM ???
        cmpb    #$7E      ;check it is present?
        bne     LFCE4      ;leave if not?
        jsr     $EFFF      ;call speech synth?
LFCE4:
        ;^
        tsta
        beq     LFD0E
        deca
        cmpa    #$0C      ;0b1100
        bhi     LFCF4      ;if A > #$0C
        jsr     LFB81      ;loads a set of values from TABLE4
        jsr     LFB E7      ;Writes $24 & $1B val to PIA_BASE
        bra     LFD0E
;
LFCF4:
        ;a > #12
        cmpa    #$1B      ;0b11011 #27
        bhi     LFD06
        ;a <= #27
        suba    #$0D      ;sub 13 from a, now a >= 0 && a <=14
        asla     ;double, a>=0 && a <= 28
        ldx     #JSR_TAB    ;start of branch table
        bsr     STEP_PNT0D    ;add A to X and pnt0D
        ldx     $00,x      ;$FD58 + ?
        jsr     $00,x      ;indirect branch
        bra     LFD0E
;
LFD06:
        suba    #$1C      ;0 <= (A-28) < 3
        jsr     Lod13      ;load buf13 from table0, ($1B always = #$FF?)
        jsr     LF83F      ;Writes $1B($$FF?) to PIA_BASE (Extra life sound?)
LFD0E:
        ldaa    flg04
        oraa    X0005      ;or with flg04 value
LFD12:
        beq     LFD12      ;loop if flg04 & $05 = 0
        clra
        staa    flg07      ;clear
        ldaa    flg04
        beq     LFD1E

```

```

        jmp     LF913
LFD1E:
        jmp     LFB34
;
;A holds value to add to add to p0Dlo
;subroutine, add a to counter, inc pnt0D when overflow
STEP_PNT0D:
LFD21:
        stx     pnt0D
        adda    p0Dlo
        staa    p0Dlo
        bcc     LFD2C
        inc     pnt0D    ;overflow
LFD2C:
        ldx     pnt0D
        rts
;END LFD21 Subroutine
;-----

;FOR DEBUGGING
NMI:
        sei
        lds     #$007F    ;setup stack pointer
        ldx     #$FFFF
        clrb
LFD37:
        adcb    $00,x
        dex
        cpx     #$F800
        bne     LFD37
        cmpb    $00,x
        beq     LFD44
        wai
                ;wait for interrupt
LFD44:
        ldaa    #$01
        jsr     L0d13    ;load buf13 from table0, offset 1
        jsr     LF83F    ;Writes $1B($FF?) to PIA_BASE (Extra Life sound?)
        ldab    $E0FA    ;speech synth ROM???
        cmpb    #$7E    ;$7E = jmp
        bne     NMI
        jsr     $E0FA    ;speech synth ROM???
        bra     NMI

        ORG $FD58 ;Branch table
JSR_TAB:
;LFD58:
        dw     $FB49 ;subroutine addresses
        dw     $F913 ;1
        dw     $FB24 ;2
        dw     $F88C ;3
        dw     $FB71 ;4
        dw     $FB1E ;5
        dw     $F8CD ;6
        dw     $F894 ;7
        dw     $F91C ;8
        dw     $F923 ;9
        dw     $F9A6 ;10
        dw     $F9D4 ;11
        dw     $F9F3 ;12
        dw     $FA44 ;13
        dw     $FA84 ;14

        ORG $FD76 ;square wave params?, gets put in buf13
TABLE0:
        db     $40 ;$13 hi delay
        db     $01 ;$14 lo delay
        db     $00 ;$15 hi change
        db     $10 ;$16 lo change
        db     $E1 ;$17 max low delay ($E1-$01)/$10 = 14 syncs?
        db     $00 ;pnt18 sync frequency
        db     $80 ;p18lo sync
        db     $FF ;$1A hi change 2
        db     $FF ;$1B amplitude
        db     $28,$01,$00,$08,$81,$02,$00,$FF,$FF;store in $13 - $1B
        db     $28,$81,$00,$FC,$01,$02,$00,$FC,$FF

```



```
db $FF,$01,$00,$18,$41,$04,$80,$00,$FF
```

```
ORG $FD9A ;loaded by LF9C2 into PIA BASE
```

```
db $8C,$5B,$B6,$40,$BF,$49,$A4,$73,$73,$A4,$49,$BF,$40,$B6,$5B,$8C
```

```
ORG $FDAA
```

```
TABLE1:
```

```
db $0C ;0- linked list, offset to next element
```

```
db $7F ;1 store in $15
```

```
db $1D ;2 loop count for LFAB3
```

```
db $0F ;3 store in $13 pointer
```

```
db $FB ;4 lo byte of $13 pointer
```

```
db $7F,$23,$0F,$15 ;$15, LFAB3, $13, $14
```

```
db $FE,$08,$50,$8B ;FDB3
```

```
;FDB7
```

```
db $88 ;0- point to end of table
```

```
db $3E ;1 $15
```

```
db $3F ;2 loop count for LFAB3
```

```
db $02 ;3 $13 pointer
```

```
db $3E ;4 lo $13 pointer
```

```
db $7C,$04,$03,$FF ;$15, LFAB3, $13, $14
```

```
db $3E,$3F,$2C,$E2 ;FDC0
```

```
db $7C,$12,$0D,$74 ;FDC4
```

```
db $7C,$0D,$0E,$41 ;FDC8
```

```
db $7C,$23,$0B,$50 ;FDCC
```

```
db $7C,$1D,$29,$F2 ;FDD0
```

```
db $7C,$3F,$02,$3E
```

```
db $F8,$04,$03,$FF
```

```
db $7C,$3F,$2C,$E2
```

```
db $F8,$12,$0D,$74
```

```
db $F8,$0D,$0E,$41
```

```
db $F8,$23,$0B,$50
```

```
db $F8,$1D,$2F,$F2
```

```
db $F8,$23,$05,$A8
```

```
db $F8,$12,$06,$BA
```

```
db $F8,$04,$07,$FF
```

```
db $7C,$37,$04,$C1
```

```
db $7C,$23,$05,$A8
```

```
db $7C,$12,$06,$BA
```

```
db $3E,$04,$07,$FF
```

```
db $3E,$37,$04,$C1
```

```
db $3E,$23,$05,$A8
```

```
db $1F,$12,$06,$BA
```

```
db $1F,$04,$07,$FF
```

```
db $1F,$37,$04,$C1
```

```
db $1F,$23,$16,$A0
```

```
db $FE,$1D,$17,$F9
```

```
db $7F,$37,$13,$06
```

```
db $7F,$3F,$08,$FA
```

```
db $FE,$04,$0F,$FF
```

```
db $FE,$0D,$0E,$41
```

```
db $FE,$23,$0B,$50
```

```
db $FE,$1D,$5F,$E4
```

```
;FE40
```

```
db $00 ;zero terminated
```

```
ORG $FE41 ;rate table
```

```
TABLE2:
```

```
db $47,$3F,$37,$30,$29,$23,$1D,$17,$12,$0D,$08,$04
```

```
ORG $FE4D ;audible waveforms
```

```
TABLE3:
```

```
db $08 ;sample length
```

```
ORG $FE4E ;hi amp small sinewave
```

```
db $7F,$D9,$FF,$D9,$7F,$24,$00,$24
```

```
ORG $FE56 ;fragmented wave
```

```
db $08 ;sample length
```

```
db $00,$40,$80,$00,$FF,$00,$80,$40
```

```
db $10 ;sample length
```

```
ORG $FE60 ;small sine
```

```
db $7F,$B0,$D9,$F5,$FF,$F5,$D9,$B0,$7F,$4E,$24,$09,$00,$09,$24,$4E
```

```
db $10 ;sample length
```

```

ORG $FE69 ;wiggly sine wave
db $7F,$C5,$EC,$E7,$BF,$8D,$6D,$6A,$7F,$94,$92,$71,$40,$17,$12,$39

```

```

db $10 ;sample length
ORG $FE82 ;max amp big square wave
db $FF,$FF,$FF,$FF,$00,$00,$00,$00,$FF,$FF,$FF,$FF,$00,$00,$00,$00

```

```

db $48 ;sample length
ORG $FE93 ;large sine
db $8A,$95,$A0,$AB,$B5,$BF,$C8,$D1,$DA,$E1,$E8,$EE,$F3,$F7,$FB,$FD
db $FE,$FF,$FE,$FD,$FB,$F7,$F3,$EE,$E8,$E1,$DA,$D1,$C8,$BF,$B5,$AB
db $A0,$95,$8A,$7F,$75,$6A,$5F,$54,$4A,$40,$37,$2E,$25,$1E,$17,$11
db $0C,$08,$04,$02,$01,$00,$01,$02,$04,$08,$0C,$11,$17,$1E,$25,$2E
db $37,$40,$4A,$54,$5F,$6A,$75,$7F

```

```

ORG $FEDB ;small sine#2
db $10 ;sample length
db $59,$7B,$98,$AC,$B3,$AC,$98,$7B,$59,$37,$19,$06,$00,$06,$19,$37

```

```

ORG $FEEC ;
TABLE4:
db $81 ;0 store in $14, and $13
db $24 ;1 hi in $15, lo in $11 = wave to subtract ($FE82 ;max amp big square wave)
db $00 ;2 store in $1A = subtraction multiplier
db $00 ;3 store in $16
db $00 ;4 store in $17
db $16 ;5 store in pnt1D + #TABLE5 (table length)
db $31 ;6 TABLE5 pitch envelope offset ($FF86 ;non-linear bigger ramp)

db $12 ;0-
db $05 ;1 ($FE93 ;large sine)
db $1A ;2 subtraction multiplier
db $FF ;3
db $00 ;4
db $27 ;5 length
db $6D ;6 TABLE5 $FFC2 ;low amp expo triangle

db $11 ;0-
db $05 ;1 ($FE93 ;large sine)
db $11 ;2 subtraction multiplier
db $01 ;3
db $0F ;4
db $01 ;5 length
db $47 ;6 TABLE5 $FF9C ;low amp small square

db $11 ;0-
db $31 ;1 ($FE56 ;fragmented wave)
db $00 ;2 subtraction multiplier
db $01 ;3
db $00 ;4
db $0D ;5 length
db $1B ;6 TABLE5 $FF70 ;low amp saw wave table

db $F4 ;0-
db $12 ;1 ($FE60 ;small sine)
db $00 ;2 subtraction multiplier
db $00 ;3
db $00 ;4
db $14 ;5 length
db $47 ;6 TABLE5 $FF9C ;low amp small square

db $41 ;0-
db $45 ;1 ($FE93 ;large sine)
db $00 ;2 subtraction multiplier
db $00 ;3
db $00 ;4
db $0F ;5 length
db $5B ;6 TABLE5 $FFB0 ;low amp ramp with 'blip'

db $21 ;0-
db $35 ;1 ($FE93 ;large sine)
db $11 ;2 subtraction multiplier
db $FF ;3
db $00 ;4
db $0D ;5 length

```

```

    db $1B ;6 TABLE5 $FF70 ;low amp saw wave table

    db $15 ;0-
    db $00 ;1 ($FE4E ;hi amp small sinewave)
    db $00 ;2 subtraction multipler
    db $FD ;3
    db $00 ;4
    db $01 ;5 length
    db $69 ;6 TABLE5 $FFBE ;???

    db $31 ;0-
    db $11 ;1 ($FE56 ;fragmented wave)
    db $00 ;2 subtraction multipler
    db $01 ;3
    db $00 ;4
    db $03 ;5 length
    db $6A ;6 TABLE5 $FFBF ;???

    db $01 ;0-
    db $15 ;1 ($FE93 ;large sine)
    db $01 ;2 subtraction multipler
    db $01 ;3
    db $01 ;4
    db $01 ;5 length
    db $47 ;6 TABLE5 $FF9C ;low amp small square

    db $F6 ;0-
    db $53 ;1 ($FE69 ;wiggly sine wave)
    db $03 ;2 subtraction multipler
    db $00 ;3
    db $02 ;4
    db $06 ;5 length
    db $94 ;6 TABLE5 $FFE9 ;very low amp saw

    db $6A ;0-
    db $10 ;1 ($FE4E ;hi amp small sinewave)
    db $02 ;2 subtraction multipler
    db $00 ;3
    db $02 ;4
    db $06 ;5 length
    db $9A ;6 TABLE5 $FFE9 ;???

    db $1F ;0-
    db $12 ;1 ($FE60 ;small sine)
    db $00 ;2 subtraction multipler
    db $FF ;3
    db $10 ;4
    db $04 ;5 length
    db $69 ;6 TABLE5 $FFBE ;sub-ramp

    db $31 ;0-
    db $11 ;1 ($FE56 ;fragmented wave)
    db $00 ;2 subtraction multipler
    db $FF ;3
    db $00 ;4
    db $0D ;5 length
    db $00 ;6 TABLE5 $FF55 ;ramp

    db $12 ;0-
    db $06 ;1 ($FEDB ;small sine#2)
    db $00 ;2 subtraction multipler
    db $FF ;3
    db $01 ;4
    db $09 ;5 length
    db $28 ;6 TABLE5 $FF7D ;low amp tri table

    ORG $FF55 ;modulation envelopes
TABLE5:
    ;ramp table - $0D long?
    db $A0,$98,$90,$88,$80,$78,$70,$68,$60,$58,$50,$44,$40

    ORG $FF63 ;hi amp expo ramp table
    db $01,$01,$02,$02,$04,$04,$08,$08,$10,$10,$30,$60,$C0,$E0,$01,$01

    ORG $FF70 ;low amp ramp table - $0D long?

```

```
db $02,$02,$03,$04,$05,$06,$07,$08,$09,$0A,$0C

ORG $FF7D ;low amp tri table - $9-$0D long?
db $80,$7C,$78,$74,$70,$74,$78,$7C,$80

ORG $FF86 ;non-linear bigger ramp - $16 long
db $01,$01,$02,$02,$04,$04,$08,$08,$10,$20,$28,$30,$00,$38,$08,$40
db $48,$50,$60,$70,$80,$A0

ORG $FF9C ;low amp small square - $1-$14 long?
db $B0,$C0,$08,$40,$08,$40,$08,$40,$08,$40,$08,$40,$08,$40,$08,$40
db $08,$40,$08,$40,$08,$40

ORG $FFB0 ;low amp ramp with 'blip' - $0F long
db $01,$02,$04,$08,$09,$0A,$0B,$0C,$0E,$0F,$10,$12,$14,$16
ORG $FFBE ;sub-ramp - $1-$4 long
db $40
ORG $FFBF ;?sub sub-ramp - $3 long
$10,$08,$01

ORG $FFC2 ;low amp expo triangle - $27 long
db $01,$01,$01,$01,$02,$02,$03,$03,$04,$04,$05,$06,$08,$0A,$0C,$10
db $14,$18,$20,$30,$40,$50,$40,$30,$20,$10,$0C,$0A,$08,$07,$06,$05
db $04,$03,$02,$02,$01,$01,$01

ORG $FFE9 ;very low amp ramp with 'blip' in middle - $6 long?
db $07,$08,$09,$0A,$0C,$08,$17
ORG $FFEF ;?? - $6 long?
db $18,$19,$1A,$1B,$1C,$00,$00
db $00

ORG $FFF8
dw $FCB6 ;IRQ
dw $F801 ;SoI
dw $FD2F ;NMI
dw $F801 ;RST
```