```
;*************************************************************************
;PLAY:
;WILL PASS AUDIO FROM LEFT CHANNEL STRAIGHT THROUGH TO
;BOTH LEFT AND RIGHT CHANNELS
;*************************************************************************


START   EQU     $40

        org     p:$0
        jmp     START

        org     p:$000c
        jsr     ssi_rx_isr      ;SSI receive data
        jsr     ssi_rx_isr      ;SSI receive data with exception
        jsr     ssi_tx_isr      ;SSI transmit data
        jsr     ssi_tx_isr      ;SSI transmit with exception

        org     p:START

        movep   #$261009,x:PLL  ;these labels are defined in the
        movep   #0000,x:BCR     ;ada_init.asm program
        ori     #3,mr           ;mask interrupts
        movec   #0,sp           ;clear hardware stack pointer
        move    #0,omr          ;operating mode 0
        move    #$40,r6         ; initialise stack pointer
        move    #-1,m6          ; linear addressing

        include 'ada_init.asm'

TONE_OUTPUT     EQU     HEADPHONE_EN+LINEOUT_EN+(4*LEFT_ATTN)+(4*RIGHT_ATTN)
TONE_INPUT      EQU     MIC_IN_SELECT+(15*MONITOR_ATTN)

        move    #TONE_OUTPUT,y0         ;set up control words
        move    y0,x:TX_BUFF_BASE+2
        move    #TONE_INPUT,y0
        move    y0,x:TX_BUFF_BASE+3

loop_1

        jset    #2,x:SSISR,*    ;wait for frame sync to pass
        jclr    #2,x:SSISR,*    ;wait for frame sync

        move    x:RX_BUFF_BASE,a        ;receive left
        move    x:RX_BUFF_BASE,b        ;receive left

        move    a,x:TX_BUFF_BASE        ;transmit left
        move    b,x:TX_BUFF_BASE+1      ;transmit right

        nop

        jmp     loop_1                  ;go back to the beginning

        include 'txrx_isr.asm'

        end
```