```
; Miscellaneous definitions
VID0    EQU     $000000 ; Video board select = 0
VID1    EQU     $001000 ; Video board select = 1
CLK2    EQU     $002000 ; Select bottom of clock driver board
CLK3    EQU     $003000 ; Select top of clock driver board
CLK4    EQU     $004000 ; Select bottom of DC bias board
CLK5    EQU     $005000 ; Select top of DC bias board
AD      EQU     $000001 ; Bit to start A/D conversion
XFER    EQU     $000002 ; Bit to transfer A/D counts into the A/D FIFO
SXMIT   EQU     $00F060 ; Transmit 4 pixels from the first coadder board

; Various delay parameters
VP_DLY  EQU     $2C0000 ; Video delay time for 3 microsec/pixel
VP_DLY_PT EQU   $B80000 ; Pass through video delay time for 10.0 microsec/pixel

DLY0    EQU     $000000 ; null delay
;DLY1   EQU     $B80000 ; LSYNC delay
DLY1    EQU     $9F0000 ; LSYNC delay  ; 10us/2 = 1/2 a cycle
DLY2    EQU     $B80000 ; CLK1,2 delay
DLY3    EQU     $FF0000 ; just another delay to play with
DLY4    EQU     $020000 ; Burst-mode delay, for subarrays (120ns)

; Put all the waveform tables in SRAM
        IF      @SCP("DOWNLOAD","HOST")
        ORG     Y:$100,Y:$100                      ; Download address
        ELSE
        ORG     Y:$100,P:APL_NUM*N_W_APL+APL_LEN+$340   ; ROM address
        ENDIF

; Define switch state bits for CLK2 = "bottom" of clock board and CLK3 = "top" of clock board,
; and CLK4 = "bottom" of bias board.

VCLK            EQU     1               ; Vertical Clock                (CLK2)
CSB             EQU     2               ; Chip Select Bar (serial)      (CLK2)
DATACLK         EQU     4               ; Data Clock (serial)           (CLK2)
DATAIN          EQU     8               ; Data In (serial)              (CLK2)
HCLK            EQU     $10             ; Horizontal (fast pixel) Clock (CLK2)
FSYNCB          EQU     $40             ; Frame Sync                    (CLK2)
READEN          EQU     $100            ; READ                          (CLK2)
RESETEN         EQU     1               ; RESET                         (CLK3)
MODECTRL1       EQU     4               ; Output Mode Control 1         (CLK3)
MODECTRL2       EQU     8               ; Output Mode Control 2         (CLK3)
TSOURCE1        EQU     $20             ; Temp Voltage source 1?        (CLK3)
TSOURCE2        EQU     $40             ; Temp Voltage source 2?        (CLK3)
LSYNCB          EQU     $400            ; Line Sync                     (CLK3)
MAINRESETB      EQU     $800            ; Main Reset Bar (registers)    (CLK3)


; set default pixel sample time (This should usually match DLY1 in waveforms.asm)
PIX_TIM DC      DLY1 ; pixel clock time, written by host computer (10us default here)

; Copy of the clocking bit definition for easy reference.
;      DC     CLK2+DLY1+VCLK+CSB+DATACLK+DATAIN+HCLK+FSYNCB+READEN
;      DC     CLK3+DLY1+RESETEN+MODECTRL1+MODECTRL2+TSOURCE1+TSOURCE2+LSYNCB+MAINRESETB


WAVE_START
      DC     WAVE_END-WAVE_START-2               ; Total number of waves in the table

; use this to stuff a dummy pix into the A2D fifo without clocking the array.
; this is for speed, as there is no pix clock delay. Use this only if reading less than the
; 2 pix fifo delay (i.e. a 1x1 pix subarray)
STUFF_THE_A2D
        DC      END_STUFF_THE_A2D-STUFF_THE_A2D-2
        DC      VID0+$100000+AD                 ; Delay needed for the
        DC      VID0                            ;   coadder to get started
        DC      VID0
END_STUFF_THE_A2D

FRAME_INIT
        DC      READ_ON-FRAME_INIT-2
        DC      CLK2+DLY1+000000+000000+000000+000000+000000+FSYNCB+000000      ; Initial clock s
```

```
et
        DC      CLK3+DLY1+000000+000000+000000+000000+000000+LSYNCB+000000      ; Initial clock s
et
        DC      CLK2+DLY1+000000+000000+000000+000000+000000+000000+000000      ; PULSE LSYNC LOW
        DC      CLK2+DLY1+000000+000000+000000+000000+000000+FSYNCB+000000      ; FSYNC HI

READ_ON
        DC      READ_OFF-READ_ON-1
        DC      CLK2+DLY0+000000+000000+000000+000000+000000+FSYNCB+READEN      ; SET READ HI

READ_OFF
        DC      VSOURCE_ON-READ_OFF-1
        DC      CLK2+DLY0+000000+000000+000000+000000+000000+FSYNCB+000000      ; SET READ LO

VSOURCE_ON
        DC      VSOURCE_OFF-VSOURCE_ON-1
        DC      CLK5+$000                                                       ; SET VSOURCE[1,2
,3,4] HI  (default)

VSOURCE_OFF
        DC      CLOCK_ROW-VSOURCE_OFF-1
        DC      CLK5+$18C                                                       ; SET VSOURCE[1,2
,3,4] LO

CLOCK_ROW
        DC      CLOCK_ROW_FAST-CLOCK_ROW-2
        DC      CLK3+DLY0+000000+000000+000000+000000+000000+000000+000000      ; PULSE LSYNC LO
 - simultaneous with VCLK to save a pulse
        DC      CLK2+DLY1+VCLK+000000+000000+000000+000000+FSYNCB+000000        ; PULSE VCLK HI
  - simultaneous with LSYNC to save apulse
        DC      CLK3+DLY0+000000+000000+000000+000000+000000+LSYNCB+000000      ; LSYNC HI
        DC      CLK2+DLY0+000000+000000+000000+000000+000000+FSYNCB+000000      ;VCLK LOW

        ; this is for burst-mode (subarrays)
CLOCK_ROW_FAST
        DC      CLOCK_ROW_RESET-CLOCK_ROW_FAST-2
        DC      CLK3+DLY0+000000+000000+000000+000000+000000+000000+000000      ; PULSE LSYNC LO
 - simultaneous with VCLK to save a pulse
        DC      CLK2+DLY4+VCLK+000000+000000+000000+000000+FSYNCB+000000        ; PULSE VCLK HI
  - simultaneous with LSYNC to save apulse
        DC      CLK3+DLY0+000000+000000+000000+000000+000000+LSYNCB+000000      ; LSYNC HI
        DC      CLK2+DLY4+000000+000000+000000+000000+000000+FSYNCB+000000      ; VCLK LO

CLOCK_ROW_RESET
        DC      CLOCK_COL_AND_READ_NO_XFER-CLOCK_ROW_RESET-2
        DC      CLK3+DLY0+000000+000000+000000+000000+000000+000000+000000      ; PULSE LSYNC LO
 - simultaneous with VCLK to save a pulse
        DC      CLK2+DLY1+VCLK+000000+000000+000000+000000+FSYNCB+000000        ; PULSE VCLK HI
  - simultaneous with LSYNC to save apulse
        DC      CLK3+DLY0+000000+000000+000000+000000+000000+LSYNCB+000000      ; LSYNC HI
        DC      CLK2+DLY1+000000+000000+000000+000000+000000+FSYNCB+000000      ; VCLK LO
        DC      CLK3+DLY1+RESETEN+000000+000000+000000+000000+LSYNCB+000000     ; Pulse RESET HI
        DC      CLK3+DLY1+000000+000000+000000+000000+000000+LSYNCB+000000      ; RESET LO


; The first waveform table here does A/D conversion without writing
;   The A/D values to the latch with XFER. The second waveform table does
;   the A/D conversion and writes the previous A/D value to the latch
;   because of the pipeline delay in the ADS937 A/D converter. Each of these
;   should be executed twice by tim.asm, as the pipeline delay is 2 pix.
CLOCK_COL_AND_READ_NO_XFER
        DC      END_CLOCK_COL_AND_READ_NO_XFER-CLOCK_COL_AND_READ_NO_XFER-2
        DC      CLK2+DLY1+000000+000000+000000+000000+000000+FSYNCB+READEN      ; HCLK LO
        DC      CLK2+DLY1+000000+000000+000000+000000+HCLK+FSYNCB+READEN        ; HCLK HI
        DC      VID0+VP_DLY+AD
        DC      VID0
END_CLOCK_COL_AND_READ_NO_XFER

CLOCK_COL_AND_READ
        DC      END_CLOCK_COL_AND_READ-CLOCK_COL_AND_READ-2
        DC      CLK2+DLY1+000000+000000+000000+000000+000000+FSYNCB+READEN      ; HCLK LO
        DC      CLK2+DLY1+000000+000000+000000+000000+HCLK+FSYNCB+READEN        ; HCLK HI
```

```
        DC      VID0+VP_DLY+AD+XFER
        DC      VID0
END_CLOCK_COL_AND_READ



; PT => Pass Through mode, wherein the coadder boards pass pixel data as
;   quickly as possible to the timing board for transmission to the host
; same as non-pt, but with a different delay. Do this once per pix.
CLOCK_COL_AND_READ_PT_NO_XFER
        DC      END_CLOCK_COL_AND_READ_PT_NO_XFER-CLOCK_COL_AND_READ_PT_NO_XFER-2
        DC      CLK2+DLY1+000000+000000+000000+000000+000000+FSYNCB+READEN      ; HCLK LO
        DC      CLK2+DLY1+000000+000000+000000+000000+HCLK+FSYNCB+READEN        ; HCLK HI
        DC      VID0+$100000+AD                         ; Delay needed for the
        DC      VID0                                    ;   coadder to get started
        DC      VID0
END_CLOCK_COL_AND_READ_PT_NO_XFER

CLOCK_COL_AND_READ_PT
        DC      END_CLOCK_COL_AND_READ_PT-CLOCK_COL_AND_READ_PT-2
        DC      CLK2+DLY1+000000+000000+000000+000000+000000+FSYNCB+READEN      ; HCLK LO
        DC      CLK2+DLY1+000000+000000+000000+000000+HCLK+FSYNCB+READEN        ; HCLK HI
        DC      VID0+$100000+AD+XFER
        DC      VID0
        DC      SXMIT
END_CLOCK_COL_AND_READ_PT



; These two waveforms tables only do A/D conversions and transfers of
;   latched data to the FIFOs. This is executed twice at the end of reading
;   the array to get the two last A/D values left over because of the
;   pipeline delay.
READ_LAST_TWO_PIXELS
        DC      END_READ_LAST_TWO_PIXELS-READ_LAST_TWO_PIXELS-2
        DC      VID0+VP_DLY     ; simply a delay, plus AD off
        DC      VID0+AD+XFER    ; AD on and transfer
        DC      VID0+VP_DLY     ; simply a delay?
        DC      VID0+VP_DLY     ; simply a delay, plus AD off
END_READ_LAST_TWO_PIXELS

SXMIT_LAST_TWO_PIXELS
        DC      END_SXMIT_LAST_TWO_PIXELS-SXMIT_LAST_TWO_PIXELS-2
        DC      VID0+VP_DLY_PT          ; simply a delay, plus AD off
        DC      VID0+$100000+AD+XFER    ; AD on and transfer
        DC      VID0                    ; AD off
        DC      SXMIT                   ; transmit to host
END_SXMIT_LAST_TWO_PIXELS


; the following are primarily for BO mode and a few other activities

; this is for advancing the pixel clock without an A2D conversion
CLOCK_ONLY
        DC      END_CLOCK_ONLY-CLOCK_ONLY-2
        DC      CLK2+DLY1+000000+000000+000000+000000+000000+FSYNCB+READEN
        DC      CLK2+DLY1+000000+000000+000000+000000+HCLK+FSYNCB+READEN
END_CLOCK_ONLY

HCLOCK_LOW
        DC      END_HCLOCK_LOW-HCLOCK_LOW-1
        DC      CLK2+DLY1+000000+000000+000000+000000+000000+FSYNCB+READEN
END_HCLOCK_LOW

; this is for advancing the pixel clock without an A2D conversion
CLOCK_ONLY_FAST
        DC      END_CLOCK_ONLY_FAST-CLOCK_ONLY_FAST-2
        DC      CLK2+DLY4+000000+000000+000000+000000+000000+FSYNCB+READEN
        DC      CLK2+DLY4+000000+000000+000000+000000+HCLK+FSYNCB+READEN
END_CLOCK_ONLY_FAST

; this is for an A2D conversion without advancing the pixel clock (with transfer)
```

```
A2D_ONLY
        DC      END_A2D_ONLY-A2D_ONLY-2
        DC      VID0+$100000+AD+XFER
        DC      VID0
        DC      SXMIT
END_A2D_ONLY

; this is for an A2D conversion without advancing the pixel clock (NO transfer)
A2D_ONLY_NO_XFER
        DC      END_A2D_ONLY_NO_XFER-A2D_ONLY_NO_XFER-2
        DC      VID0+$100000+AD
        DC      VID0
        DC      VID0
END_A2D_ONLY_NO_XFER

; this just pulses the rest hi and then low again
PULSE_RESET
        DC    END_PULSE_RESET-PULSE_RESET-2
        DC      CLK3+DLY1+RESETEN+000000+000000+000000+000000+LSYNCB+000000     ; Pulse RESET HI
        DC      CLK3+DLY1+000000+000000+000000+000000+000000+LSYNCB+000000      ; RESET LO
END_PULSE_RESET

; these are for the serial interface
CSB_OFF
        DC      END_CSB_OFF-CSB_OFF-1
        DC      CLK2+DLY1+000000+CSB+000000+000000+000000+FSYNCB+000000         ; SET CSB LO
END_CSB_OFF

CSB_ON
        DC      END_CSB_ON-CSB_ON-2
        DC      CLK2+DLY1+000000+CSB+000000+000000+000000+FSYNCB+000000         ; hold CSB LO for
 half a cycle
        DC      CLK2+DLY1+000000+000000+000000+000000+000000+FSYNCB+000000      ; then SET CSB HI
END_CSB_ON

CLOCK_SERIAL_ONE
        DC      END_CLOCK_SERIAL_ONE-CLOCK_SERIAL_ONE-2
        DC      CLK2+DLY1+000000+CSB+000000+DATAIN+000000+FSYNCB+000000         ; CSB LO, DATAIN
HI
        DC      CLK2+DLY1+000000+CSB+DATACLK+DATAIN+000000+FSYNCB+000000        ; CSB LO, DATAIN
HI, DATACLK HI
        DC      CLK2+DLY0+000000+CSB+000000+DATAIN+000000+FSYNCB+000000         ; CSB LO, DATAIN
HI, DATACLK LO
END_CLOCK_SERIAL_ONE

CLOCK_SERIAL_ZERO
        DC      END_CLOCK_SERIAL_ZERO-CLOCK_SERIAL_ZERO-2
        DC      CLK2+DLY1+000000+CSB+000000+000000+000000+FSYNCB+000000         ; CSB LO, DATAIN
LO
        DC      CLK2+DLY1+000000+CSB+DATACLK+000000+000000+FSYNCB+000000        ; CSB LO, DATAIN
LO, DATACLK HI
        DC      CLK2+DLY0+000000+CSB+000000+000000+000000+FSYNCB+000000         ; CSB LO, DATAIN
LO, DATACLK LO
END_CLOCK_SERIAL_ZERO
WAVE_END


; Values for IR clock and bias boards. Both boards are jumpered for bipolar operation
MAX_V    EQU     7.5             ; Maximum voltage from bias driver board
MAX_V2   EQU    15.0             ; 2 x MAX_V
CLK_ZERO EQU     0.0             ; Zero volts for power-on sequence



;  Zero out ALL the DC biases and clocks during the power-on sequence (zero all lines)
ZERO_BIASES
        DC      END_ZERO_BIASES-ZERO_BIASES-1
        ;       clock board
        DC      (CLK2<<8)+(0<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK2<<8)+(1<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
```

```
DC      (CLK2<<8)+(2<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(3<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(4<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(5<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(6<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(7<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(8<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(9<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(10<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(11<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(12<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(13<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(14<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(15<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(16<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(17<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(18<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(19<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(20<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(21<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(22<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(23<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(24<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(25<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(26<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(27<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(28<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(29<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(30<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(31<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(32<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(33<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(34<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(35<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(36<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(37<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(38<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(39<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(40<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(41<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(42<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(43<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(44<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(45<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(46<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK2<<8)+(47<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)

; bias board
DC      (CLK4<<8)+(0<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK4<<8)+(1<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK4<<8)+(2<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK4<<8)+(3<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK4<<8)+(4<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK4<<8)+(5<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK4<<8)+(6<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK4<<8)+(7<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK4<<8)+(8<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK4<<8)+(9<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK4<<8)+(10<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK4<<8)+(11<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK4<<8)+(12<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK4<<8)+(13<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK4<<8)+(14<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK4<<8)+(15<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK4<<8)+(16<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK4<<8)+(17<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK4<<8)+(18<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK4<<8)+(19<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK4<<8)+(20<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK4<<8)+(21<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK4<<8)+(22<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
DC      (CLK4<<8)+(23<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
```

```
        DC      (CLK4<<8)+(24<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK4<<8)+(25<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK4<<8)+(26<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK4<<8)+(27<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK4<<8)+(28<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK4<<8)+(29<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK4<<8)+(30<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK4<<8)+(31<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK4<<8)+(32<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK4<<8)+(33<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK4<<8)+(34<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK4<<8)+(35<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK4<<8)+(36<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK4<<8)+(37<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK4<<8)+(38<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK4<<8)+(39<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK4<<8)+(40<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK4<<8)+(41<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK4<<8)+(42<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK4<<8)+(43<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK4<<8)+(44<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK4<<8)+(45<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK4<<8)+(46<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK4<<8)+(47<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
END_ZERO_BIASES


; Initialize all DACs


; Clocking voltage definitions
;       set voltages using CLK2 0-47, just as if setting biases on CLK4, but remember that
;       switch states for output lines 0-11 are CLK2 and
;       switch states for output lines 12-23 are CLK3
;       Also, set any unused pins to zero here
CLK_HI       EQU        3.1     ; High Clock voltage
CLK_LO       EQU        0.1     ; Low Clock voltage


; Bias voltage definitions
;       The DC bias board is strapped
;       for bipolar outputs to accomodate the +1.0 volt VssOUT required for the video
;       source followers. The rail voltage is MAX_V = 7.5 volts. MAX_V2 = 2 x MAX_V = 15.0
;       Also, set any unused pins to zero here
VSOURCE1        EQU     3.0     ;
VSOURCE2        EQU     3.0     ;
VSOURCE3        EQU     3.0     ;
VSOURCE4        EQU     3.0     ;
VBIASGATE       EQU     2.4     ; pixel source-follower bias voltage
CELLDRAIN       EQU     0.0     ; pixel source-follower drain node
;DSUB           EQU     0.4     ; Detector substrate voltage
DSUB            EQU     2.20    ; Detector substrate voltage for warm mux
VDDA            EQU     3.28    ; Analog positive power supply
;VRESET         EQU     0.15    ; Detector reset voltage (actual from leach is 0.1)
VRESET          EQU     0.40    ; Detector reset voltage (actual from leach is 0.1) for warm mux
DRAIN           EQU     0.0     ; output source-follower drain node
VBIASPOWER      EQU     3.28    ; pixel source-follower source node
VDD             EQU     3.28    ; Digital positive power supply


; Video offsets
;       DAC settings for the video offsets with two coadder boards installed
;       Recommend OFFSET not greater than $FD0 and not less than $000

;OFFSET  EQU     $F00
;OFFSET  EQU     $7FF    ; use this for H1R MUX (jumpered vid board)
OFFSET  EQU     $C90    ; use this for H1RG SCA
;OFFSET  EQU     $FD0    ; use this for no det, harnesses, jumpered vid board
```

```
        ; These constants define the slope and intercept of the DACs in
        ; output ADU as a function of the above OFFSET
        ; Appropriate for IDTL coaddder boards at normal operating temp.

PP0_0   EQU     32865.699
PP1_0   EQU     32820.581
PP2_0   EQU     32689.120
PP3_0   EQU     32771.884
PP4_0   EQU     32982.305
PP5_0   EQU     32846.121
PP6_0   EQU     32559.233
PP7_0   EQU     32674.982

PP0_1   EQU     -16.021943
PP1_1   EQU     -16.056462
PP2_1   EQU     -16.082675
PP3_1   EQU     -15.948860
PP4_1   EQU     -16.122941
PP5_1   EQU     -16.119988
PP6_1   EQU     -16.037983
PP7_1   EQU     -16.057203

        ; balance all amps to amp 5, assuming the given OFFSET
OFFSET0 EQU     @CVI((((OFFSET*PP0_1)+PP0_0)-((OFFSET*PP6_1)+PP6_0))/(PP0_1*(-1)))+OFFSET
OFFSET1 EQU     @CVI((((OFFSET*PP1_1)+PP1_0)-((OFFSET*PP6_1)+PP6_0))/(PP1_1*(-1)))+OFFSET
OFFSET2 EQU     @CVI((((OFFSET*PP2_1)+PP2_0)-((OFFSET*PP6_1)+PP6_0))/(PP2_1*(-1)))+OFFSET
OFFSET3 EQU     @CVI((((OFFSET*PP3_1)+PP3_0)-((OFFSET*PP6_1)+PP6_0))/(PP3_1*(-1)))+OFFSET
OFFSET4 EQU     @CVI((((OFFSET*PP4_1)+PP4_0)-((OFFSET*PP6_1)+PP6_0))/(PP4_1*(-1)))+OFFSET
OFFSET5 EQU     @CVI((((OFFSET*PP5_1)+PP5_0)-((OFFSET*PP6_1)+PP6_0))/(PP5_1*(-1)))+OFFSET
OFFSET6 EQU     @CVI((((OFFSET*PP6_1)+PP6_0)-((OFFSET*PP6_1)+PP6_0))/(PP6_1*(-1)))+OFFSET
OFFSET7 EQU     @CVI((((OFFSET*PP7_1)+PP7_0)-((OFFSET*PP6_1)+PP6_0))/(PP7_1*(-1)))+OFFSET



DACS    DC      END_DACS-DACS-1

                ; remember, odd #'s are "0" state (default), and even #'s are "1" state
                                                                        ; These are the DB37 pin
IDs
CLOCKS
        DC      (CLK2<<8)+(0<<14)+@CVI((CLK_HI+MAX_V)/MAX_V2*4095)      ; Pin #1 VCLK
        DC      (CLK2<<8)+(1<<14)+@CVI((CLK_LO+MAX_V)/MAX_V2*4095)
        DC      (CLK2<<8)+(2<<14)+@CVI((CLK_LO+MAX_V)/MAX_V2*4095)
        DC      (CLK2<<8)+(3<<14)+@CVI((CLK_HI+MAX_V)/MAX_V2*4095)      ; Pin #2 CSB (def HI)
        DC      (CLK2<<8)+(4<<14)+@CVI((CLK_HI+MAX_V)/MAX_V2*4095)      ; Pin #3 DATACLK
        DC      (CLK2<<8)+(5<<14)+@CVI((CLK_LO+MAX_V)/MAX_V2*4095)
        DC      (CLK2<<8)+(6<<14)+@CVI((CLK_HI+MAX_V)/MAX_V2*4095)      ; Pin #4 DATAIN
        DC      (CLK2<<8)+(7<<14)+@CVI((CLK_LO+MAX_V)/MAX_V2*4095)
        DC      (CLK2<<8)+(8<<14)+@CVI((CLK_HI+MAX_V)/MAX_V2*4095)      ; Pin #5 HCLK
        DC      (CLK2<<8)+(9<<14)+@CVI((CLK_LO+MAX_V)/MAX_V2*4095)
        DC      (CLK2<<8)+(10<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)   ; Pin #6
        DC      (CLK2<<8)+(11<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK2<<8)+(12<<14)+@CVI((CLK_HI+MAX_V)/MAX_V2*4095)     ; Pin #7 FSYNCB
        DC      (CLK2<<8)+(13<<14)+@CVI((CLK_LO+MAX_V)/MAX_V2*4095)
        DC      (CLK2<<8)+(14<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)   ; Pin #8
        DC      (CLK2<<8)+(15<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK2<<8)+(16<<14)+@CVI((CLK_HI+MAX_V)/MAX_V2*4095)     ; Pin #9 READEN
        DC      (CLK2<<8)+(17<<14)+@CVI((CLK_LO+MAX_V)/MAX_V2*4095)
        DC      (CLK2<<8)+(18<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)   ; Pin #10
        DC      (CLK2<<8)+(19<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK2<<8)+(20<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)   ; Pin #11
        DC      (CLK2<<8)+(21<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK2<<8)+(22<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)   ; Pin #12
        DC      (CLK2<<8)+(23<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK2<<8)+(24<<14)+@CVI((CLK_HI+MAX_V)/MAX_V2*4095)     ; Pin #13 RESETEN
        DC      (CLK2<<8)+(25<<14)+@CVI((CLK_LO+MAX_V)/MAX_V2*4095)
        DC      (CLK2<<8)+(26<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)   ; Pin #14
        DC      (CLK2<<8)+(27<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK2<<8)+(28<<14)+@CVI((CLK_HI+MAX_V)/MAX_V2*4095)     ; Pin #15 MODECTRL1
        DC      (CLK2<<8)+(29<<14)+@CVI((CLK_LO+MAX_V)/MAX_V2*4095)
        DC      (CLK2<<8)+(30<<14)+@CVI((CLK_HI+MAX_V)/MAX_V2*4095)     ; Pin #16 MODECTRL2
        DC      (CLK2<<8)+(31<<14)+@CVI((CLK_LO+MAX_V)/MAX_V2*4095)
```

```
        DC      (CLK2<<8)+(32<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)    ; Pin #17
        DC      (CLK2<<8)+(33<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK2<<8)+(34<<14)+@CVI((CLK_HI+MAX_V)/MAX_V2*4095)      ; Pin #18 TSOURCE1
        DC      (CLK2<<8)+(35<<14)+@CVI((CLK_LO+MAX_V)/MAX_V2*4095)
        DC      (CLK2<<8)+(36<<14)+@CVI((CLK_HI+MAX_V)/MAX_V2*4095)      ; Pin #19 TSOURCE2
        DC      (CLK2<<8)+(37<<14)+@CVI((CLK_LO+MAX_V)/MAX_V2*4095)
        DC      (CLK2<<8)+(38<<14)+@CVI((CLK_LO+MAX_V)/MAX_V2*4095)
        DC      (CLK2<<8)+(39<<14)+@CVI((VDD+MAX_V)/MAX_V2*4095)         ; Pin #33 (BIAS) VDD
        DC      (CLK2<<8)+(40<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)    ; Pin #34
        DC      (CLK2<<8)+(41<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK2<<8)+(42<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)    ; Pin #35
        DC      (CLK2<<8)+(43<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)
        DC      (CLK2<<8)+(44<<14)+@CVI((CLK_HI+MAX_V)/MAX_V2*4095)      ; Pin #36 LSYNCB
        DC      (CLK2<<8)+(45<<14)+@CVI((CLK_LO+MAX_V)/MAX_V2*4095)
        DC      (CLK2<<8)+(46<<14)+@CVI((CLK_HI+MAX_V)/MAX_V2*4095)      ; Pin #37 MAINRESETB
        DC      (CLK2<<8)+(47<<14)+@CVI((CLK_LO+MAX_V)/MAX_V2*4095)
END_CLOCKS

                                                                ; These are the DB37 pin IDs
BIASES
        DC      (CLK4<<8)+(0<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)            ;
        DC      (CLK4<<8)+(1<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)            ; Pin #1
        DC      (CLK4<<8)+(2<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)            ;
        DC      (CLK4<<8)+(3<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)            ; Pin #2
        DC      (CLK4<<8)+(4<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)            ;
        DC      (CLK4<<8)+(5<<14)+@CVI((VRESET+MAX_V)/MAX_V2*4095)              ; Pin #3 VRESET
        DC      (CLK4<<8)+(6<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)            ;
        DC      (CLK4<<8)+(7<<14)+@CVI((DSUB+MAX_V)/MAX_V2*4095)                ; Pin #4 DSUB
        DC      (CLK4<<8)+(8<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)            ;
        DC      (CLK4<<8)+(9<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)            ; Pin #5
        DC      (CLK4<<8)+(10<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ;
        DC      (CLK4<<8)+(11<<14)+@CVI((VBIASGATE+MAX_V)/MAX_V2*4095)          ; Pin #6 VBIASGAT
E
        DC      (CLK4<<8)+(12<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ;
        DC      (CLK4<<8)+(13<<14)+@CVI((DRAIN+MAX_V)/MAX_V2*4095)              ; Pin #7 DRAIN
        DC      (CLK4<<8)+(14<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ;
        DC      (CLK4<<8)+(15<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ; Pin #8
        DC      (CLK4<<8)+(16<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ;
        DC      (CLK4<<8)+(17<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ; Pin #9
        DC      (CLK4<<8)+(18<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ;
        DC      (CLK4<<8)+(19<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ; Pin #10
        DC      (CLK4<<8)+(20<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ;
        DC      (CLK4<<8)+(21<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ; Pin #11
        DC      (CLK4<<8)+(22<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ;
        DC      (CLK4<<8)+(23<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ; Pin #12
        DC      (CLK4<<8)+(24<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ;
        DC      (CLK4<<8)+(25<<14)+@CVI((VBIASPOWER+MAX_V)/MAX_V2*4095)         ; Pin #13 VBIASPO
WER
        DC      (CLK4<<8)+(26<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ;
        DC      (CLK4<<8)+(27<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ; Pin #14
        DC      (CLK4<<8)+(28<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ;
        DC      (CLK4<<8)+(29<<14)+@CVI((VSOURCE2+MAX_V)/MAX_V2*4095)           ; Pin #15 VSOURCE
2
        DC      (CLK4<<8)+(30<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ;
        DC      (CLK4<<8)+(31<<14)+@CVI((VSOURCE4+MAX_V)/MAX_V2*4095)           ; Pin #16 VSOURCE
4
        DC      (CLK4<<8)+(32<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ;
        DC      (CLK4<<8)+(33<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ; Pin #17
        DC      (CLK4<<8)+(34<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ;
        DC      (CLK4<<8)+(35<<14)+@CVI((CELLDRAIN+MAX_V)/MAX_V2*4095)          ; Pin #18 CELLDRA
IN
        DC      (CLK4<<8)+(36<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ;
        DC      (CLK4<<8)+(37<<14)+@CVI((VDDA+MAX_V)/MAX_V2*4095)               ; Pin #19 VDDA
        DC      (CLK4<<8)+(38<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ;
        DC      (CLK4<<8)+(39<<14)+@CVI((VSOURCE1+MAX_V)/MAX_V2*4095)           ; Pin #33 VSOURCE
1
        DC      (CLK4<<8)+(40<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ;
        DC      (CLK4<<8)+(41<<14)+@CVI((VSOURCE3+MAX_V)/MAX_V2*4095)           ; Pin #34 VSOURCE
3
        DC      (CLK4<<8)+(42<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ;
        DC      (CLK4<<8)+(43<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ; Pin #35
        DC      (CLK4<<8)+(44<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)           ;
```

```
        DC      (CLK4<<8)+(45<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)          ; Pin #36
        DC      (CLK4<<8)+(46<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)          ;
        DC      (CLK4<<8)+(47<<14)+@CVI((CLK_ZERO+MAX_V)/MAX_V2*4095)          ; Pin #37
END_BIASES

OFFSETS
        DC     $0c0000+OFFSET0   ; Input offset board #0, channel A coadder     ; OFFSET0
        DC     $0c4000+OFFSET1   ; Input offset board #0, channel B coadder     ; OFFSET1
        DC     $0c8000+OFFSET2   ; Input offset board #0, channel C coadder     ; OFFSET2
        DC     $0cc000+OFFSET3   ; Input offset board #0, channel D coadder     ; OFFSET3
        DC     $1c0000+OFFSET4   ; Input offset board #1, channel A coadder     ; OFFSET4
        DC     $1c4000+OFFSET5   ; Input offset board #1, channel B coadder     ; OFFSET5
        DC     $1c8000+OFFSET6   ; Input offset board #1, channel C coadder     ; OFFSET6
        DC     $1cc000+OFFSET7   ; Input offset board #1, channel D coadder     ; OFFSET7
END_OFFSETS

END_DACS


END_WAVEFORMS
```