

FMUTIL.H

```

/*****
/*          F M U T I L . H          */
/**-----**/
/* Task      : Header file for FMUTIL.C */
/**-----**/
/* Author     : Michael Tischer / Bruno Jennrich */
/* Developed on : 03/20/1994 */
/* Last update : 04/05/1995 */
/**-----**/
/* COMPILER   : Borland C++ 3.1, Microsoft Visual C++ 1.5 */
/*****
#define __INC_FM_UTIL_H

#include "types.h"
#include "sbutil.h"

#define ADLIB_PORT      0x388
#define AL_TIMER1       0x02
#define AL_TIMER2       0x03
#define AL_TISTATE      0x04

#define AL_TI1STARTSTOP 0x01
#define AL_TI2STARTSTOP 0x02
#define AL_TI2MASK      0x20
#define AL_TI1MASK      0x40
#define AL_TIRESET      0x80

/* Bits of status register */
#define AL_STAT2OVRFLW  0x20 /* Timer 2 overflow */
#define AL_STAT1OVRFLW  0x40 /* Timer 1 overflow */
#define AL_STATIRQ      0x80

```

```

#define SB_REGPORT      0                      /* Register port */
#define SB_STATUSPORT  0                      /* Status port */
#define SB_DATAPORT     1                      /* Data port */

/* Oscillator numbers of percussion instruments */
#define BASSDRUM_A      12                    /* takes frequency from Channel 6 */
#define HIHAT           13                    /* takes frequency from Channel 7 */
#define TOMTOM          14                    /* takes frequency from Channel 8 */
#define BASSDRUM_B      15                    /* takes frequency from Channel 6 */
#define SNAREDRUM       16                    /* takes frequency from Channel 7 */
#define TOPCYMBAL       17                    /* takes frequency from Channel 8 */

#define BASS_CHANNEL     6
#define HIHATSNARE_CHANNEL 7
#define TOMTOMCYMBAL_CHANNEL 8

#define FM_FIRSTOPL2 0
#define FM_SECNDOPL2 1

/* Frequency parameters of default scale */
/* FrqParam = 1.13 * Frq (0-1023) */
#define _c 343                                /* 262 Hz */
#define _cis 363                             /* 277 Hz */
#define _d 385                               /* 294 Hz */
#define _dis 408                             /* 311 Hz */
#define _e 432                               /* 330 Hz */
#define _f 458                               /* 349 Hz */
#define _fis 485                             /* 370 Hz */
#define _g 514                               /* 392 Hz */
#define _gis 544                             /* 415 Hz */
#define _a 577                               /* 440 Hz */
#define _ais 611                             /* 466 Hz */

```

```

#define _h      647                                /* 494 Hz */
#define _c2     686                                /* 523 Hz */

/* Tone duration as ten-thousandth of a time constant */
#define _1_1    10000    /* whole tone      ( 10000 / 1 ) */
#define _1_2    5000     /* half tone       ( 10000 / 2 ) */
#define _1_4    2500     /* quarter tone    ( 10000 / 4 ) */
#define _1_8    1250     /* eighth tone     ( 10000 / 8 ) */
#define _1_16   625      /* sixteenth tone  ( 10000 / 16 ) */
#define _1_32   312      /* thirty-second tone ( 10000 / 32 ) */

/* Macro for calculating actual duration of tone */
/* base = Duration of a whole tone in thousandths of a second */
#define CalcDelay( base, d ) ( ( int ) ( ( long )base * ( long )d ) \
    / _1_1 ) )

INT fm_Write( INT iOpl, INT iReg, BYTE iVal );
INT fm_WriteBit( INT iOpl, INT iReg, INT iBit, BYTE bSet );
VOID fm_Reset( VOID );
VOID fm_SetBase( PSBBASE pSBBASE, INT iReset );
INT fm_GetAdLib( PSBBASE pSBBASE );
INT fm_GetChannel( INT o_nr );
INT fm_GetModulator( INT ch_nr );
INT fm_GetCarrier( INT ch_nr );
VOID fm_SetOscillator( INT iOpl, INT iOsc, BYTE bA, BYTE bD, BYTE bS,
    BYTE bR, BYTE bShortADSR, BYTE bContADSR,
    BYTE bVibrato, BYTE bTremolo, BYTE bMute,
    BYTE bHiMute, BYTE bFrqFactor, BYTE bWave );
VOID fm_SetModulator( INT iOpl, INT iChn, BYTE bA, BYTE bD, BYTE bS,
    BYTE bR, BYTE bShortADSR, BYTE bContADSR,
    BYTE bVibrato, BYTE bTremolo, BYTE bMute,
    BYTE bHiMute, BYTE bFrqFactor, BYTE bWave );

```

```

VOID fm_SetCarrier( INT iOpl, INT iChn, BYTE bA, BYTE bD, BYTE bS,
                    BYTE bR, BYTE bShortADSR, BYTE bContADSR,
                    BYTE bVibrato, BYTE bTremolo, BYTE bMute,
                    BYTE bHiMute, BYTE bFrqFactor, BYTE bWave );
VOID fm_SetChannel( INT iOpl, INT iChn, BYTE bOct, INT iFrq, BYTE bFM,
                    BYTE bFeedBack );
VOID fm_SetCard      ( INT iOpl, BYTE bVibrato, BYTE bTremolo );
VOID fm_PlayChannel  ( INT iOpl, INT iChn, BYTE bOn );
VOID fm_PlayHiHat    ( INT iOpl, BYTE bOn );
VOID fm_PlayTopCymbal ( INT iOpl, BYTE bOn );
VOID fm_PlayTomTom    ( INT iOpl, BYTE bOn );
VOID fm_PlaySnareDrum ( INT iOpl, BYTE bOn );
VOID fm_PlayBassDrum  ( INT iOpl, BYTE bOn );
VOID fm_PercussionMode( INT iOpl, BYTE bOn );
VOID fm_PollTime      ( LONG lMilli );
INT  fm_QuadroOn( INT iOn );
VOID fm_ChannelLR( INT iOpl, INT iChn, INT iL, INT iR );
VOID fm_QuadroChannel( INT iOpl, INT iCH0, INT iCH1, INT iCH2 );
VOID fm_QuadroMode( INT iOpl, INT iChn, INT iMode );

#endif

```