# Bankstick Routines

This routines will load/save patterns from a bankstick module

We're using the writepage MIOS function because a write to the bankstick takes a long time this way we can write 64bytes at once so only 4 eeprom accesses are necessary

First versions of the firmware had a pattern size of 128bytes so a pattern could be saved with two calls to the writepage function

Saving a pattern with 256bytes takes four page writes, this makes it impossible to save the four pages after each other without affecting the sequencer timing.
To avoid this the pattern is written over four ticks. Each time the program cycles through the main loop a block is written until the whole pattern is saved.

Reading from a eeprom is a lot quicker so we can just call readpage for each block...

Code from bankstick.c:

```
/////////////////////////////////////////////////////////////////////////////
/
//  This function initialises a pattern save
/////////////////////////////////////////////////////////////////////////////
/
void BS_SavePat (unsigned char pat_nr){
    //only init block save if no other save is in progress
    if (save_block_cnt < NR_BLOCK)
        return;

    //each pattern contains 256 bytes
    //so address = pat_nr*256
    writeaddr = ((unsigned int)pat_nr) << 8;
    save_block_cnt = 0;
}
/////////////////////////////////////////////////////////////////////////////
/
//  This function saves a pattern block to a bankstick
/////////////////////////////////////////////////////////////////////////////
/
void BS_SaveBlock (){
    unsigned char error = 0;
    //check if something has to be saved
    if (save_block_cnt < NR_BLOCK) {
        //write block
        error |= MIOS_BANKSTICK_WritePage(writeaddr, buffer +
(save_block_cnt<<6));
        //calculate address of next block
        writeaddr += 0x40;
        save_block_cnt++;
```

```
        if (error) {
            MIOS_LCD_CursorSet(0xC0 + 0);
            MIOS_LCD_PrintCString("error saving pattern");
            MIOS_LCD_MessageStart(128);
            //stop saving pattern
            save_block_cnt = NR_BLOCK;
        }
    }

}

/////////////////////////////////////////////////////////////////////////////
/
//  This function loads a pattern from a bankstick
/////////////////////////////////////////////////////////////////////////////
/
void BS_LoadPat (unsigned char pat_nr){
    unsigned int readaddr;
    unsigned char i;
    //each pattern contains 256 bytes
    //so address = pat_nr*256
    readaddr = ((unsigned int)pat_nr) << 8;
    //load all values to bankstick
    //(will be optimised in the future)
    MIOS_BANKSTICK_ReadPage(readaddr + 0x00, buffer + 0x00);
    MIOS_BANKSTICK_ReadPage(readaddr + 0x40, buffer + 0x40);
    MIOS_BANKSTICK_ReadPage(readaddr + 0x80, buffer + 0x80);
    MIOS_BANKSTICK_ReadPage(readaddr + 0xC0, buffer + 0xC0);

    MIOS_LCD_CursorSet(0xC0 + 0);
    if (MIOS_BOX_STAT.BS_AVAILABLE) {
        MIOS_LCD_PrintCString("load ok");
    } else {
        MIOS_LCD_PrintCString("load error");
    }
    MIOS_LCD_MessageStart(128);
}
```

back to DSEQ32

From:
https://midibox.org/dokuwiki/ - **MIDIbox**

Permanent link:
**https://midibox.org/dokuwiki/doku.php?id=dseq_bs**

Last update: **2006/12/12 12:10**