HayBall

Daniel Panero (EL-342800)

Systèmes Embarqués Microprogrammés

NDS features: checklist (1/3)

- **ARM Processors**
  - ARM9 to control the two screens, the arrows and START keys, load the music stream from the SD-Card and send IPC messages to the ARM7 to control the sound. ARM7 to control the touchscreen and the audio
- **Timers / Interrupts**
  - Timer 0 with 64 divider to animate the sprites and background shifts
  - Timer 2 to manage the sound stream
- **Graphics**
  - Main screen: Mode 5 2D with Background 2 (Ext. Rotoscale) and VRAM A
  - Sub screen: Mode 0 2D with Background 0,1 (Tiled) and VRAM C
- **Keypad**
  - Interrupt waiting for START for starting the game
  - Polling to play the game either with the touch screen or with the arrow keys
  - The tractor's sprite moves accordingly to the arrow key you are pressing

NDS features: checklist (2/3)

- **Touchscreen**
  - Polling to play the game either with the touch screen or with the arrow keys.
  - The tractor's sprite follows you where you touched (and smoothly turns around)
- **Sound**
  - Custom music stream handler to manage the music/sound effects
  - Streaming of music/sound effects from the SD-Card, allowing for long and accurate music in wav file (3 minutes, approx. 21 mb)
- **Secondary Storage (optional)**
  - Used to store music/sound effect in the sound folder and stream from it
- **Sprites (optional)**
  - Several sprites in the oamSub were used with complex rotations and animations

NDS features: checklist (3/3)

- **WiFi (optional)**
  - Not used

Music stream (1/2)

```
static mm_stream *musicStream;

static struct Sound soundMusic = {.samplingRate = 44100, .format = MM_STREAM_16BIT_MONO};
static struct Sound soundEffectWin = {.samplingRate = 44100, .format = MM_STREAM_16BIT_MONO};
static struct Sound soundEffectLost = {.samplingRate = 44100, .format = MM_STREAM_16BIT_MONO};

void soundInit()
{
    soundMusic.file = fopen("./sound/music.wav", "rb");
    ...

    sassert(soundMusic.file, "File music not found!");
    ...


    mm_ds_system sys;
    sys.mod_count = 0;
    sys.samp_count = 0;
    sys.mem_bank = 0;
    sys.fifo_channel = FIFO_MAXMOD;
    mmInit(&sys);

    musicStream->sampling_rate = 44100;
    musicStream->buffer_length = 20000;
    musicStream->callback = fillMusicStream;
    musicStream->format = MM_STREAM_16BIT_MONO;
    musicStream->timer = MM_TIMER2;
    musicStream->manual = 1;

    mmSetModuleVolume(512);

    mmStreamOpen(musicStream);
}
```

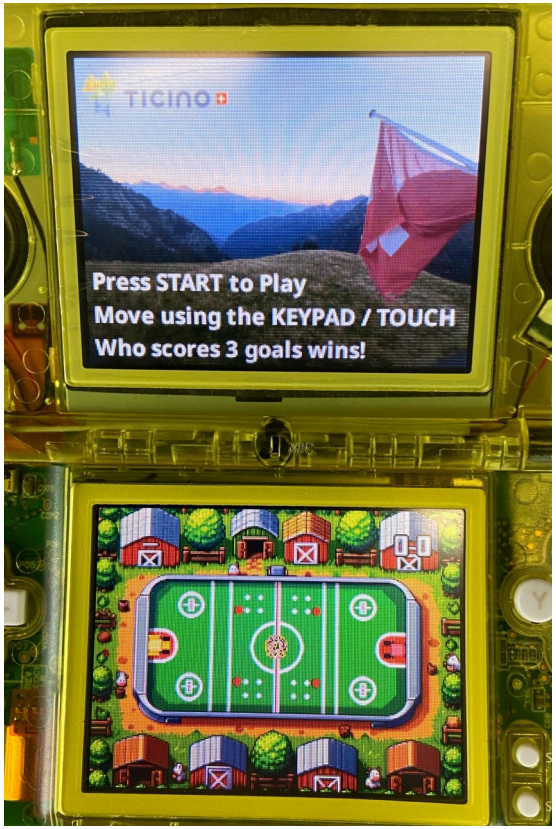
Music stream (2/2)

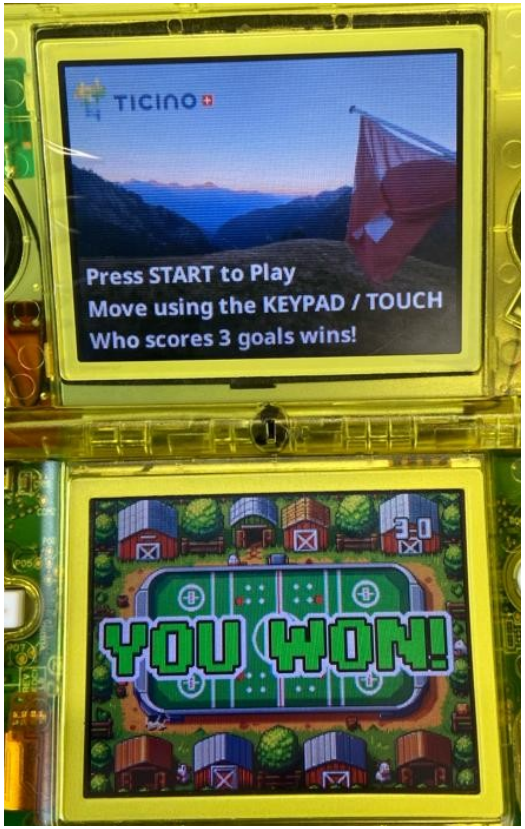
```
mm_word fillMusicStream(mm_word length, mm_addr dest, mm_stream_formats format)
{
    int res;

    switch (state)
    {
    case MUSIC:
        res = fread(dest, 4, length, soundMusic.file);
        if (res != length)
        {
            rewind(soundMusic.file);
            length = res - (res % 4);
        }
        else
        {
            length = res;
        }
        break;

    case EFFECT:
    {
      ...
    }
    break;
    }
    return length;
}
```

NDS project screenshots





Notable features and future ideas

- **Notable features**
  - The collision boxes are circular (therefore a more nuanced physics)
  - Friction, different masses and collision are based on physics
  - Sprites, Sound are managed using structs so they can be easily added

- **Future ideas**
  - Extending sound.c in its own independent library: automatic discovery of music and sound effect, replicate Maxmod sound distortion (panning…), multichannel playing...

Let’s play!