
# GPL951XX SPU API Programming Guide

*V0.1 – Aug. 02, 2017*

***Preliminary***

No.19, Industry E. Rd. IV, Hsinchu Science Park, Hsinchu City 30077, Taiwan, R.O.C

Tel: 886-3-666-2118 Fax: 886-3-666-2117 Web: www.generalplus.com

**Important Notice**

Generalplus Technology reserves the right to change this documentation without prior notice. Information provided by Generalplus Technology is believed to be accurate and reliable. However, Generalplus Technology makes no warranty for any errors which may appear in this document. Contact Generalplus Technology to obtain the latest version of device specifications before placing your order. No responsibility is assumed by Generalplus Technology for any infringement of patent or other rights of third parties which may result from its use. In addition, generalplus products are not authorized for use as critical components in life support devices/systems or aviation devices/systems, where a malfunction or failure of the product may reasonably be expected to result in significant injury to the user, without the express written approval of GENERALPLUS.

# Table of Content

# Revision History

## Document History

| Revision | Date | By | Remark |
|---|---|---|---|
| V1.0 | 08/02/2017 | Frank Kung | Original Version |

## Library History

| Revision | Date | By | Remark |
|---|---|---|---|
| V1.0.4 | 03/13/2017 | Frank Kung | GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib |

# 1 SPU Library Summary

**SPU Driver:**

| Present Algorithm Title | Application |
|---|---|
| SPU Driver | SPU Driver |

SPU drive is mainly designed to play midi music and sound effect.

# 2 Service loop

Rather than using ordinary software implementation to generate musical tones, SPU driver utilizes the hardware approach to produce clear and high quality tones along with initiative interrupt mechanism to execute the service loop.

**Example:**

**Background service loop:**

**In main.c**

```
int main()
{
        System_Initial();                                   // System initial
        InitSPU();
        SetMidiChannelMask(0x0000,0xFFFF);
        SetMidiVolume(127);
        p = &MIDI_Table;
        PlayMidi(*(p+1), *p, SS_PLAYMIDI_ONCE);
       while(1)
        {
              System_ServiceLoop();                         // Service loop for watchdog clear
        }
        return 0;
}
```

**in isr.asm:**

```
_FIQ:
        call    _F_IRQ4_Service                 // SPU Driver ISR Service Loop
```

# 3 Memory Allocation

In SPU driver, it is necessary to use certain RAM blocks to decode Midi sequences. The RAM spaces used for midi sequence decode can be shared with user's program by aligning the RAM blocks manually. The memory allocation list can be found in the file, project_name.map, inside the directory, release or debug. The u'nSP IDE (v1.6 or later) also provides programmers a convenient tool, memory map, which graphically lists all the memory used by each module, section, public function, and variable.

The principle of sharing RAM is to guarantee the RAM block can be shared through ORAM or OSRAM section declaration as long as the algorithms or applications are not active simultaneously. For more information about the RAM/ORAM/OSRAM, please refer to Generalplus u'nSP Assembly Tools User's Manual.

If user chooses more than one algorithm in the same project, but the program is not going to run more than one algorithm at the same time, the advantage of ORAM section is to allow programmer sharing the same physical memory block among different algorithms. Programmer can use u'nSP IDE (Project→ Setting→ section) to align the ORAM address. If programmer is not satisfied with the manual allocation and would prefer returning to default compiler arrangement, the only thing he or she to do is deleting the file, project_name.lik, in project folder and rebuild all projects again. The memory allocation will be realigned based on default compiler rules.

The RAM block section definitions are as follows:

Table: Name of RAM in the library

| RAM definition | | |
|---|---|---|
| **Algorithm** | **RAM Label** | **Size(word)** |
| SPU Driver | SPU_DRV_RAM_BLOCK | 390 (0x186) |

# 4 API of SPU-Library

## 4.1 Function: Initialize SPU module

**Syntax:**

**C:** Int InitSPU (void)

**ASM:** N/A

**Parameters:** None.

**Return Value:** SPU driver version

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** initializes SPU

## 4.2 Function: Set Midi Channel Mask

**Syntax:**

**C:** int SetMidiChannelMask (unsigned short iHMask, unsigned short iLMask)

**ASM:** N/A

**Parameters:** **iHMask:** MASK word bits from B0 to B15 represent channel 0 to channel 15

**iLMask:** MASK word bits from B0 to B15 represent channel 16 to channel 31

**Return Value:** None.

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** When a bit equals 1, the corresponding channel will be used by MIDI play algorithm. In contrast, the corresponding channel will not be allocated for MIDI play when bit eauals to 0.

User can reserve some channels for playing CELP or SPEECH

## 4.3 Function: Set Midi Volume

**Syntax:**

**C:** int SetMidiVolume (short iVolume)

**ASM:** N/A

**Parameters:** iVolume: 0 … 127 to set MIDI playback volume

**Return Value:** None.

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Set MIDI volume dynamically

## 4.4 Function: Set Midi Tempo

**Syntax:**

**C:** int SetMidiTempo (unsigned int MidiTempo)

**ASM:** N/A

**Parameters:** MIDI Tempo value you want to set.

**Return Value:** None.

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Set MIDI tempo dynamically

## 4.5 Function: Initial Midi stop call back function

**Syntax:**

**C:** void InitMidiStopCallBackFunc (void *Func())

**ASM:** N/A

**Parameters:** Call back function address pointer in bank 0

**Return Value:** None.

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Initial the call back function when stop playing will call to.

## 4.6 Function: Disable Midi stop call back function

**Syntax:**

**C:** void DisableMidiStopCallBackFunc (void)

**ASM:** N/A

**Parameters:** None

**Return Value:** None

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Disables auto MIDI end Call back

## 4.7 Function: Play Midi

**Syntax:**

**C:** void PlayMidi (iSeg, iOffset, iPlayMode)

**ASM:** N/A

**Parameters:** iSeg - Segment of sequence data that is compiled by SunMidiar

iOffset - Offset of sequence data

iPlayMode - SS_PLAYMIDI_ONCE | SS_PLAYMIDI_INFINITY

SS_PLAYMIDI_ONCE: Play MIDI once

SS_PLAYMIDI_INFINITY: Play MIDI repeatedly

**Return Value:** None.

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Plays a specified MIDI sequence compiled by SunMidiar

## 4.8 Function: Stop Midi

**Syntax:**

**C:** void StopMidi (void)

**ASM:** N/A

**Parameters:** None.

**Return Value:** None.

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Stops Midi

## 4.9 Function: Pause Midi

**Syntax:**

**C:** void PauseMidi (void)

**ASM:** N/A

**Parameters:** None.

**Return Value:** None.

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Pauses MIDI

## 4.10 Function: ResumeMidi

**Syntax:**

**C:** int ResumeMidi (void)

**ASM:** N/A

**Parameters:** None.

**Return Value:** None.

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Resumes Midi

## 4.11 Function: Detect FIFO Overflow

**Syntax:**

**C:** void Detect_FIFO_Overflow (void)

**ASM:** N/A

**Parameters:** None

**Return Value:** None

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Detects if FIFO is overflow. If yes (overflow), it will call toggle function.

## 4.12 Function: Get Low Channel Status

**Syntax:**

**C:** int GetChannelStatus (void)

**ASM:** N/A

**Parameters:** None.

**Return Value:** Status register word

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Gets low channel status register. Each bit in the return status register stands for one channel

All bits in register word mean, 1: busy, 0: idle.

## 4.13 Function: Get High Channel Status

**Syntax:**

**C:** int GetChannelStatus_H (void)

**ASM:** N/A

**Parameters:** None.

**Return Value:** Status register word

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Gets high channel status register. Each bit in the return status register stands for one channel

All bits in register word mean, 1: busy 0: idle.

## 4.14 Function: Get SPU Status

**Syntax:**

**C:** int GetSPUStatus (void)

**ASM:** N/A

**Parameters:** None.

**Return Value:** R_SPU_Status word

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Gets SPU Status

## 4.15 Function: Check Channel Is Busy or Idle

**Syntax:**

**C:** void IsChannelStop (unsigned short iChannelNo)

**ASM:** N/A

**Parameters:** The channel number from 0 to 15.

**Return Value:** 1 - The channel is idle or stopped

0 - The channel is busy

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Checks whether the specified channel is busy or idle

## 4.16 Function: Query Midi Original Tempo

**Syntax:**

**C:** int QueryMidiOriginalTempo (void)

**ASM:** N/A

**Parameters:** None.

**Return Value:** MIDI original tempo value.

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Acquires Midi original tempo.

## 4.17 Function: Query Midi Current Tempo

**Syntax:**

**C:** int QueryMidiCurrentTempo (void)

**ASM:** N/A

**Parameters:** None.

**Return Value:** MIDI current Tempo value

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Acquires Midi current tempo.

## 4.18 Function: System Reset Midi Tempo

**Syntax:**

**C:** void sysResetMidiTempo (void)

**ASM:** N/A

**Parameters:** None.

**Return Value:** None.

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Resets MIDI tempo to its initial state.

The MIDI tempo will be initialized automatically while playing a new MIDI.

This function is needed when MIDI tempo shall be initialized while midi is playing.

## 4.19 Function: Play a PCM(Drum)

**Syntax:**

**C:** void TM_PlayPCM (int iSeg, int iOffset, int iPan, int iVelocity)

**ASM:** N/A

**Parameters:** iSeg - Segment of specified DRM file

iOffset - Offset of specified DRM file

iPan - Balance value PAN

iVelocity - Amplitude of playing channel from 0 to maximum 127

**Return Value:** Allocated channel number for this DRM file

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Plays a specified DRM file on a random channel with 4-bit ADPCM or 8/16-bit PCM mode.

## 4.20 Function: Play a PCM (Drum) with a Fixed Channel

**Syntax:**

**C:** void TM_PlayPCM_FixCH (int iChannelNo, int iSeg, int iOffset, int iPan, int iVelocity)

**ASM:** N/A

**Parameters:** iChannelNo - Specifies channel to play PCM/ADPCM

iSeg - Segment of specified DRM file

iOffset - Offset of specified DRM file

iPan - Balance value PAN

iVelocity - Amplitude of playing channel from 0 to maximum 127

**Return Value:** Playing channel number for this DRM file

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Plays a specified DRM file on a specified channel with 4-bit ADPCM or 8/16-bit PCM mode

## 4.21 Function: Play a PCM (Drum) from SFX library

**Syntax:**

**C:** void TM_PlayPCM_Lib (int iIndex, int iPan, int iVelocity)

**ASM:** N/A

**Parameters:** iIndex –ToneColor Index in SFX library

iPan - Balance value PAN

iVelocity - Amplitude of playing channel from 0 to maximum 127

**Return Value:** Playing channel number for this DRM file

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Plays a specified DRM file on a random channel with 4-bit ADPCM or 8/16-bit PCM mode from SFX library.

## 4.22 Function: Play a PCM (Drum) with a fixed Channel from SFX library

**Syntax:**

**C:** void TM_PlayPCM_FixCH_Lib (int iChannelNo, int iIndex, int iPan, int iVelocity)

**ASM:** N/A

**Parameters:** iChannelNo - Specified channel for playing PCM/ADPCM

iIndex - ToneColor Index in SFX library

iPan - Balance value PAN

iVelocity - Amplitude of playing channel from 0 to maximum 127

**Return Value:** Playing channel number for this DRM file

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Plays a specified DRM file on a specified channel with 4-bit ADPCM or 8/16-bit PCM mode from SFX library.

## 4.23 Function: Play a Loop PCM (Drum) with a fixed Channel from SFX library

**Syntax:**

**C:** void TM_PlayLoopPCM_FixCH_Lib (int iChannelNo, int iIndex, int iPan, int iVelocity)

**ASM:** N/A

**Parameters:** iChannelNo - Specified channel for playing PCM/ADPCM

iIndex - ToneColor Index in SFX library

iPan - Balance value PAN

iVelocity - Amplitude of playing channel from 0 to maximum 127

**Return Value:** Playing channel number for this DRM file

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Circularly plays a specified DRM file on a specified channel with 4-bit ADPCM or 8/16-bit PCM mode from SFX library.

## 4.24 Function: Play a PCM (Drum) without envelope from SFX library

**Syntax:**

**C:** void TM_PlayPCM_NoEnv_Lib (int iIndex, int iPan, int iVelocity)

**ASM:** N/A

**Parameters:** iIndex –ToneColor Index in SFX library

iPan - Balance value PAN

iVelocity - Amplitude of playing channel from 0 to maximum 127

**Return Value:** Playing channel number for this DRM file

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Plays a specified DRM file without envelope on a random channel with 4-bit ADPCM or 8/16-bit PCM mode from SFX library.

## 4.25 Function: Play a PCM (Drum) without envelope with a fixed Channel from SFX library

**Syntax:**

**C:** void TM_PlayPCM_NoEnv_FixCH_Lib (int iChannelNo, int iIndex, int iPan, int iVelocity)

**ASM:** N/A

**Parameters:** iChannelNo - Specified channel for playing PCM/ADPCM

iIndex - ToneColor Index in SFX library

iPan - Balance value PAN

iVelocity - Amplitude of playing channel from 0 to maximum 127

**Return Value:** Playing channel number for this DRM file

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Plays a specified DRM file without envelope on a specified channel with 4-bit ADPCM or 8/16-bit PCM mode from SFX library.

## 4.26 Function: Play a Tone (Alp)

**Syntax:**

**C:** void TM_PlayTone (int PlayPitch, int iSeg, int iOffset, int iPan, int iVelocity)

**ASM:** N/A

**Parameters:** PlayPitch - Pitch number of playing note

iSeg - Segment of specified ALP file

iOffset - Offset of specified ALP file

iPan - Balance value PAN

iVelocity - Amplitude of playing channel from 0 to maximum 127

**Return Value:** Playing channel number for this ALP file

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Plays a specified ALP file with variable pitches, as well as PAN and velocity

## 4.27 Function: Play a Tone(Alp) with a Fixed Channel

**Syntax:**

**C:** void TM_PlayTone_FixCH (int iChannelN, int PlayPitch, int iSeg, int iOffset, int iPan, int iVelocity)

**ASM:** N/A

**Parameters:** iChannelNo: Specified channel for playing note

PlayPitch - Pitch number of playing note

iSeg - Segment of specified ALP file

iOffset - Offset of specified ALP file

iPan - Balance value PAN

iVelocity - Amplitude of playing channel from 0 to maximum 127

**Return Value:** Playing channel number for this ALP file

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Plays a specified ALP file with variable pitches, as well as PAN and velocity with specified channel.

## 4.28 Function: Stop a Playing Channel

**Syntax:**

**C:** void TM_StopPCM (int iChannelN)

**ASM:** N/A

**Parameters:** iChannelNo: Specified channel for stop playing

**Return Value:** None.

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Stops a specified playing channel

## 4.29 Function: Get SPU driver version

**Syntax:**

**C:** int Get_SPU_DrvVersion (void)

**ASM:** N/A

**Parameters:** None.

**Return Value:** version number

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Obtains the SPU driver version.

## 4.30 Function: Stop SPU

**Syntax:**

**C:** void StopSPU(void);

**ASM:** N/A

**Parameters:** None

**Return Value:** None

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** Disable all SPU channels.

## 4.31 Function: Interrupt service routine for SPU driver

**Syntax:**

**C:** N/A

**ASM:** F_IRQ4_Service

**Parameters:** None

**Return Value:** None

**Library:** <GLB_GP-FS1_0405A_95100_SPU_1.0.4.lib>

**Remark:** SPU Driver Entry Point

# 5 Resources List of SPU Driver

## 5.1 TABLE 1: RAM Size (Unit: Decimal Word)

| | IRAM | ISRAM | RAM | SRAM | ORAM | OSRAM |
|---|---|---|---|---|---|---|
| SPU Driver | | | | | 390 | |

## 5.2 TABLE 2: ROM Size (Unit: Decimal Word)

| | TEXT | CODE | DATA | USER DEFINE |
|---|---|---|---|---|
| SPU Driver | | 5487 | 2465 | |

## 5.3 TABLE 3: Hardware Resources VS Library

| | Interrupt | Timer Setting | Audio |
|---|---|---|---|
| SPU Driver | IRQ4 | 5ms | DAC |

## 5.4 TABLE 4: CPU Usage Rate (approximate)

| | CPU Usage Rate (54 MHz) while SPU Driver |
|---|---|
| SPU Driver | 30% (1.5ms/5ms) |

## 5.5 TABLE 5: Name of RAM in the library

Table: Name of RAM in the library

| Overlap RAM definition | | |
|---|---|---|
| **Algorithm** | **RAM Label** | **Size(word)** |
| SPU Driver | SPU_DRV_RAM_BLOCK | 390 (0x186) |