SNDS pSOS for pNA+ Loopback test

I. pSOS Build and Execute flow

II. Return Setup For Diagnostic Program

III. Setup pSOS Source Code & Make File For SNDS

pSOS Build and Execute flow

1. **Edit C:\AUTOEXEC.BAT**

```
REM #####   Setup for pSOSystem
SET ARMLIB=C:\ARM211\LIB\Embedded
SET ARMINC=C:\ARM211\INCLUDE
SET PSS_ROOT=c:\user\psos\snds\rev0.0
SET PATH=%PSS_ROOT%\Bin\Win32;%PATH%
SET PSS_BSP=%PSS_ROOT%\bsps\5000
SET BSP_TYPE=32b
```

2. **Invoke ARM Project Manager and Change Option**
   1) Tools -> Configure -> armlink
   2) go **Entry and Base** page
   3) erase **Read-Write** area
   4) go **ImageLayout** page
   5) erase **Object**(init.o) File and **Area Name**(init)
   6) Select **Ok** button

3. **System Reboot**

4. **Make Directory**
   C:\USER\pSOS\SNDS\REV0.0

5. **Extraxt pSOS.ZIP to C:\USER\pSOS\SNDS\REV0.0 Directory**

6. **Change Directory and Build OS Library**

```
CD %PSS_ROOT%\SYS\OS
PMAKE
```

7. **Change Directory and Build BSP Library**

```
CD %PSS_ROOT%\BSPS\5000\SRC
PMAKE
```

8. **Change Directory and Build Application Program Image**

```
CD %PSS_ROOT%\APPS\LOOPBACK
PMAKE
```

9. **Download and Execute**

```
SFTP 1 RAM.BIN
```

Setup For Diagnostic Program

1. **Invoke ARM Project Manager and Change Option (Not necessory for command line build)**
   1) Tools -> Configure -> armlink
   2) go **Entry and Base** page
   3) erase **Read-Write** area : 0x1300000
   4) go **ImageLayout** page
   5) write **Object**(init.o) File and **Area Name**(init)
   6) Select **Ok** button

2. **ARM Library Directory Set**

```
SET ARMLIB=C:\ARM211\LIB
```

Setup pSOS Source Code & Make File For SNDS

1. Edit %PSS_ROOT%\BSPS\5000\SRC\INIT.S

```
.....

;---------------------------------------------------------------
; Setup SWI Vector Table
;---------------------------------------------------------------
        LDR   r0, =0x13FFFD8    ; SWI exception table address
        LDR   r1, =SystemSwiHandler

.....

;---------------------------------------------------------------------
; Set the SVC mode (we are not in user mode, but may be in other non-SVC
; mode) and disable interrupts. Note, we do not disable FIQ.
;---------------------------------------------------------------------
SvcStart
 [ BOOT_ROM
        MRS     r0, cpsr
        AND     r0, r0, #PSR_Mode_Mask
        ORR     r0, r0, #PSR_I_Bit+PSR_F_Bit+PSR_SVC_Mode
        MSR     cpsr, r0
 |
        swi     0xff    ; SVC Change
 ]

.....

;---------------------------------------------------------------------
; System SWI Handler
;---------------------------------------------------------------------
SystemSwiHandler
        STMFD   sp!, {r0-r12, lr}
        LDR   r0, [lr, #-4]
        BIC   r0, r0, #0xff000000
        CMP   r0, #0xff
        BEQ   MakeSVC
        LDMFD   sp!, {r0-r12, pc}^

MakeSVC
        MRS   r1, spsr
        BIC   r1, r1, #ModeMask
        ORR   r2, r1, #SVC32Mode
        MSR   spsr, r2
        LDMFD   sp!, {r0-r12, pc}^
```

2. Edit %PSS_ROOT%\BSPS\DEVICES\ARM\VECTORS.S

```
.......

SoftVectors     EQU  0x13FFFD0

.......
```

3. Edit %PSS_ROOT%\SYS\OS\MAKEFILE

```
.......
        @$(ECHO) "#########################################################"
        @$(ECHO) "#  Building ARM  LITTLE  ENDIAN  OS    Libraries          #"
        @$(ECHO) "#########################################################"
        @pmake  -f  make32l.mk  libsys.32l
        @pmake  -f  make32l.mk  libsysxx.32l
.......
        @$(ECHO) "#########################################################"
        @$(ECHO) "#  Building ARM  BIG  ENDIAN  OS    Libraries             #"
        @$(ECHO) "#########################################################"
        @pmake  -f  make32b.mk libsys.32b
        @pmake  -f  make32b.mk libsysxx.32b
.......
        @$(ECHO) "#########################################################"
        @$(ECHO) "#  Building THUMB  LITTLE  ENDIAN  OS    Libraries       #"
        @$(ECHO) "#########################################################"
        @pmake  -f  make16l.mk  libsys.16l
        @pmake  -f  make16l.mk  libsysxx.16l
.......
        @$(ECHO) "#########################################################"
        @$(ECHO) "#  Building THUMB  BIG  ENDIAN  OS    Libraries          #"
        @$(ECHO) "#########################################################"
        @pmake  -f  make16b.mk libsys.16b
        @pmake  -f  make16b.mk libsysxx.16b
.......
```

4. Edit %PSS_ROOT%\CONFIGS\STD\CONFIG.MK

```
.....

ram.bin: c.opt a.opt dir lnkram.tmp sys_conf.h $(RAM_SYS_OBJS) \
        $(PSS_DRVOBJS) $(PSS_COMPLIB) \
        $(DRV_LIB1) $(DRV_LIB2) $(DRV_LIB3) $(DRV_LIB4) $(DRV_LIB5)
        @cat lnkram.tmp                                  > lnkram.cmd
        @$(ECHO) $(RAMOPTS)                               >> lnkram.cmd
        @$(ECHO) $(LDOPTS2) ram.map                       >> lnkram.cmd
        $(LD) $(DEBUG) -bin -o ram.bin -VIA lnkram.cmd
        @$(RM) *.opt
        @$(RM) lnkram.cmd
        @$(RM) lnkram.tmp
.....
```

5. Edit %PSS_ROOT%\APPS\LOOPBACK\MAKEFILE

```
.....

PSS_BSP          =          $(PSS_ROOT)/bsps/5000
.....
```

5. Edit %PSS_BSP%\BSP.MK

```
RAMOPTS          = -Base 0x1000050
OSOPTS           = -Base 0x1000050

APPBASE          = 0x1000050
APPOPTS          = -Base $(APPBASE)
```