## Anexo VI: DTC_DS

```
;////////////////////////////////////////////////////////////////////////////////////////
; PROJECT INFORMATION:
;        Name:                              pol_dtc_00.s
;        Author                             J. Granado
;        Date of creation           March 4th, 1999
;        Date of last review                March 9th, 1999
;        Version                            1.0 not a version
;        Purpose:                   to extract information about SBM
;
;////////////////////////////////////////////////////////////////////////////////////////
; SUBROUTINES INCLUDED:
;        IRQ_Handler
;        RESET_HANDLER
;        POL_DTC_00_main
;        UART_00_IAR
;        UART_00_ICR
;        TIMER_00_IAR
;        TIMER_00_ICR
;
;////////////////////////////////////////////////////////////////////////////////////////
; HARDWARE HYPOTESIS
;        Based on: Policom Data Sheet of December 1998
;
;////////////////////////////////////////////////////////////////////////////////////////
; COMMENTS:
;        SWI Handler not installed
;        FIQ Handler not installed
;        IRQ Handler uses its hidden features
;        Dot forget to program the EMC main parameters
;
;////////////////////////////////////////////////////////////////////////////////////////
;  RESULTS:
;        This sotfware will allow you to have an interface between PC-DTC
;        which could help you to test the DTC.
;        It supports messgaes interchange between PC and DTC which includes
;        remote configuration of SMB and Timer parameters.
;////////////////////////////////////////////////////////////////////////////////////////

             AREA    rom, CODE, READONLY


;*********************************************************
;                     GENERAL MEMORY MAP
;*********************************************************

;-------------------------- RAM-----------------------------------------------------------
RAM_INT_Base    EQU     0x100000                                ; RAM: from 100_000 to 102_000
RAM_INT_Limit   EQU     RAM_INT_Base + RAM_SIZE - 4
RAM_SIZE EQU      0x2000                                        ; 2 x 4 KBytes as Internal Ram

;-------------------------- ROM-----------------------------------------------------------
ROM_INT_Limit      EQU      0x10000                    ; Internal ROM limit
                                                                ; internal ROM located at 0x000000
; --------------------------stacks ----------------------------------------------------------
-
STACK_SIZE         EQU      400                                ; 1 KByte as stack size
IRQ_Stack       EQU     RAM_INT_Limit                 ;  IRQ stack at top of memory
SVC_Stack       EQU     IRQ_Stack-STACK_SIZE          ; followed by SVC stack
USR_Stack       EQU         SVC_Stack-STACK_SIZE      ; followed by USR stack
STACK_Limit        EQU      USR_Stack-STACK_SIZE

;-------------------------------- Interrupt vector and priority Tables-----------------------------------------
-------
EXCEPTION_VECTOR_Base         EQU     RAM_INT_Base
EXCEPTION_VECTOR_Limit        EQU     EXCEPTION_VECTOR_Base+(IRQ_SOURCES_NUMBER-1)*4
;*************************************************************************
EXCEPTION_PRIORITY_Base       EQU     AIC_Base+0x300
;EXCEPTION_PRIORITY_Limit EQU         EXCEPTION_PRIORITY_Base+(IRQ_SOURCES_NUMBER-1)
;        ***************************************
;        Remember that this version does not implement
;        the PVT into the RAM.
;        ***************************************

;*************************************************************************
SWI_VECTOR_Base               EQU     EXCEPTION_VECTOR_Limit+4 ;
SWI_VECTOR_Limit    EQU     SWI_VECTOR_Base + 4*32 -4
;
;
;-------------------------- special addresses-----------------------------------------------------------
---------
```

```
RESERVED_MEMORY_Base        EQU     SWI_VECTOR_Limit+4
RESERVED_MEMORY_Limit       EQU     RESERVED_MEMORY_Base+4*32-4
;
;-------------------------- external memory-------------------------------------------------------
---------
ROM_EXT_Base      EQU     0x200000
RAM_EXT_Base      EQU     0x300000
;
;
;-------------------------- User data for general purposes ----------------------------------------
---------
INT_RAM_USER_DATA_Base      EQU     RESERVED_MEMORY_Limit+4
INT_RAM_USER_DATA_Limit EQU STACK_Limit-4
;
;--------------------------------
;-       variables          -
;--------------------------------

Mode_USR        EQU     0x10
Mode_IRQ         EQU     0x12
Mode_SVC        EQU     0x13

I_Bit            EQU     0x80
F_Bit           EQU     0x40



;*********************************************************
;                            PERIPHERAL MAP
;        Based on in Data-sheet (Dec.98 version)
;
;*********************************************************

;-------------------------- ASB map ---------------------------

EMC_Base  EQU     0x400000
AIC_Base  EQU     0X410000
APB_Base  EQU     0x420000

;-------------------------- APB map----------------------------
UART_Base EQU     0x420000
BPP_Base  EQU     0x430000
TIMER_Base        EQU     0x460000
SBM_Base  EQU     0x480000
WD_Base           EQU     0x470000

;--------------------------E M C---------------------------
emc_ROMCF EQU     EMC_Base
emc_RAMCF EQU     EMC_Base+0x4


;--------------------------A I C ---------------------------                       -
aic_IRS           EQU     AIC_Base
aic_ISS           EQU     AIC_Base+0x4
aic_IER           EQU     AIC_Base+0x8
aic_IES           EQU     AIC_Base+0x8
aic_IEC           EQU     AIC_Base+0xC
aic_SIT           EQU     AIC_Base+0x10
aic_FSS           EQU     AIC_Base+0x100
aic_FRS           EQU     AIC_Base+0x104
aic_FER           EQU     AIC_Base+0x108
aic_FES           EQU     AIC_Base+0x108
aic_FEC           EQU     AIC_Base+0x10C
aic_PRS           EQU     AIC_Base+0x200
;--------------------------------
;    AIC BIT IDENTIFICATION
;--------------------------------

FIQ               EQU     0x0
COMMRX            EQU     0x2
COMMTX            EQU     0x3
TIMER0            EQU     0x4
TIMER1            EQU     0x5
SER_A             EQU     0x8
BPP               EQU     0xA
SBM               EQU     0xC
SPI               EQU     0xD

IRQ_SOURCES_NUMBER EQU 20 ; tiene que ser multiplo de 4


;--------------------------UART----------------------------
uart_RBR  EQU     UART_Base+0
uart_THR  EQU     UART_Base+0
uart_DLL  EQU     UART_Base+0
uart_IER  EQU     UART_Base+1
uart_DLM  EQU     UART_Base+1
```

```
uart_IIR EQU      UART_Base+2
uart_FCR EQU      UART_Base+2
uart_LCR EQU      UART_Base+3
uart_MCR EQU      UART_Base+4
uart_LSR EQU      UART_Base+5
uart_MSR EQU      UART_Base+6
uart_SCR EQU      UART_Base+7

;---------------------------B P P----------------------------
bpp_IAMR          EQU      BPP_Base
bpp_IMR           EQU      BPP_Base+0x4
bpp_ISR           EQU      BPP_Base+0x8
bpp_IAR           EQU      BPP_Base+0xC
bpp_IIMR EQU      BPP_Base+0x10
bpp_IIECREQU      BPP_Base+0x14
bpp_IIDCREQU      BPP_Base+0x18
bpp_IOLR EQU      BPP_Base+0x1c
bpp_OLR           EQU      BPP_Base+0x20

;---------------------------T I M E R----------------------------
timer_TC0_IR      EQU      TIMER_Base+0x0
timer_TC0_CV      EQU      TIMER_Base+0x4
timer_TC0_CI      EQU      TIMER_Base+0x8
timer_TC0_CR      EQU      TIMER_Base+0xC
timer_TC1_IR      EQU      TIMER_Base+0x20
timer_TC1_CV      EQU      TIMER_Base+0x24
timer_TC1_CI      EQU      TIMER_Base+0x28
timer_TC1_CR      EQU      TIMER_Base+0x2C
timer_TC2_IR      EQU      TIMER_Base+0x40
timer_TC2_CV      EQU      TIMER_Base+0x44
timer_TC2_CI      EQU      TIMER_Base+0x48
timer_TC2_CR      EQU      TIMER_Base+0x4C


;--------------------------- S B M ----------------------------
sbm_RBR_0EQU      SBM_Base
sbm_RBR_1EQU      SBM_Base+0x04
sbm_IIR  EQU      SBM_Base+0x08
sbm_IER  EQU      SBM_Base+0x0C
sbm_MOD_0EQU      SBM_Base+0x10
sbm_MOD_1EQU      SBM_Base+0x14
sbm_LSR_0EQU      SBM_Base+0x20
sbm_LSR_1EQU      SBM_Base+0x24
sbm_NSE_00        EQU      SBM_Base+0x28
sbm_NSE_01        EQU      SBM_Base+0x2C
sbm_NSE_10        EQU      SBM_Base+0x30
sbm_NSE_11        EQU      SBM_Base+0x34
sbm_CCR  EQU      SBM_Base+0x50
sbm_BDT  EQU      SBM_Base+0x54
sbm_BPT  EQU      SBM_Base+0x58
sbm_THR  EQU      SBM_Base+0x5C

;---------------------------W D----------------------------
wd_CV             EQU      WD_Base+0x0
wd_CCV   EQU      WD_Base+0x0
wd_LR             EQU      WD_Base+0x4
wd_ER             EQU      WD_Base+0x8


;*********************************************************
;                    AMM_POL_DTC_00
;                  APPLICATION MEMORY MAP
;*********************************************************




;*********************************************************
;                           ARM CODE
;        Begins with the exceoption vectors
;*********************************************************


        ENTRY
;---------------------------initial vector table (BEGIN)----------------------------
        B       Reset_Handler
        B       Reset_Handler
        B       Reset_Handler
        B       Reset_Handler
        B       Reset_Handler
        NOP                             ; Reserved vector
        B       IRQ_Handler
        B       Reset_Handler

;---------------------------initial vector table (END)----------------------------
```

```
;*****************************************************
; SUBROUTINE INFORMATION:
;        Name:                      RESET_HANDLER_ver_4
;        Author:                    J. Granado
;        Date of creation  Dec. 16, 1998
;        Last review                Mar 6,1999
;        Version                    ver 4.0
;        Comments:         this version has been done to
;                                   to be included in POL_DTC_00
;
;*****************************************************
; SUBROUTINE PARAMEMTERS: NONE
;
;*****************************************************
; EXECUTION CONTEXT
;        Reset state.
;
;*****************************************************
; LABELS
;
;
;*****************************************************



Reset_Handler
Reset_Handler_ver_3_Begin

;++++++++++++++++++++++++++++++++
; ARM state:       IRQ disable
;                  FIQ disable
;                  Supervision Mode
;++++++++++++++++++++++++++++++++
         ;  Register used:
         ;        R0  -> data
         ;        R10 -> bpp output level register

;---------------------------
; ---> Initialise registers
;---------------------------

         mov      r0,#0
         mov      r1,#0
         mov      r2,#0
         mov      r3,#0
         mov      r4,#0
         mov      r5,#0
         mov      r6,#0
         mov      r7,#0
         mov      r8,#0
         mov      r9,#0
         mov      r10,#0
         mov      r11,#0
         mov      r12,#0
         mov      r13,#0
         mov      r14,#0

;-----------------------------------------------------
; Externa Memory Controller
;-----------------------------------------------------

         LDR      R6,=emc_ROMCF
         LDR      R7,=emc_RAMCF
         MOV      R0, #0x330
         STR      R0, [R6]
         MOV      R0, #0x025
         STR      R0, [R7]

;-----------------------------------------------------
;        Ready to external execution
;-----------------------------------------------------

;-----------------------------------------------------
; Enter IRQ mode and set up the IRQ stack pointer
;---------------------------;-------------------------
       MOV     R0, #Mode_IRQ:OR:I_Bit:OR:F_Bit ; No interrupts
       MSR     CPSR, R0
       LDR     R13, =IRQ_Stack

;-------------------------------------------------------------
; Set up the SVC stack pointer last and return to SVC mode
;-------------------------------------------------------------
      MOV     R0, #Mode_SVC:OR:I_Bit:OR:F_Bit ; No interrupts
      MSR     CPSR, R0
      LDR     R13, =SVC_Stack

;-----------------------------------------------
```

```
;      Disable watchdog
;------------------------------------------------
        LDR     R0, =wd_ER
        MOV     R1,#0x0
        STR     R1,[R0]

;++++++++++++++++++++++++++++++++++++++++++++
; ARM STATE:      - IRQ disabled and FIQ enabled
;                 - supervision mode
;++++++++++++++++++++++++++++++++++++++++++++

;-------------------------------------------------------------------------------
;     Now check and clear Internal RAM
;-------------------------------------------------------------------------------
        ;  Register used:
        ;       R0  -> Internal ram base addr
        ;       R1  -> not used
        ;       R2  -> 0000
        ;       R3  -> fail counter
        ;       R4  -> not used
        ;       R5  -> value read from ram
        ;       R6  -> int ram test ok msg
        ;       R7  -> int ram test failed msg
        ;       R9  -> counter
        ;       R10 -> ram test flag addr
        ;       R11 -> RAM LIMIT


        LDR     R0, =RAM_INT_Base
        MOV     R2, #0x0
        MOV     R3, #0
        MOV     R9, #0x0
        LDR     R11, =RAM_INT_Limit

Reset_Handler_ver_3_label_3
        LDR     R5,[R0]
        STR     R2, [R0], #4
        ADD     R9, R9, #1
        CMP     R0, R11
        BLE     Reset_Handler_ver_3_label_3

;       WD was not enabled
        BL      USER_CODE; jump to user code


;++++++++++++++++++++++++++++++++++++++++++++
; ARM STATE:      - IRQ disabled and FIQ enabled
;                 - supervision MODE
;++++++++++++++++++++++++++++++++++++++++++++

;******(END of ROUTINE)*****************************







;*****************************************************
; SUBROUTINE INFORMATION:
;       Name:                   IRQ_Handler
;       Author:                 J. Granado
;       Date of creation        July 7, 1998
;       Last review (4.0)  Nov. 4, 1998
;       Version                 ver 4.0
;       Comments:
;                tested (ARMulator + Verilog )
;                compatible with software aic implementation
;                (POLICOM DIGITAL SPECIFICATIONS ver 7.0, Nover, 1998)
;                - Hardware Priority Implementation (hidden feratures)
;
;*****************************************************
; SUBROUTINE PARAMEMTERS: NONE
;
;*****************************************************
; EXECUTION CONTEXT
;                - SPSR contains old CPSR
;                - IRQ's are disabled
;                - LR contains return PC
;                - ARM mode
;
;*****************************************************
; LABELS
;
;
;*****************************************************
IRQ_Handler
```

```
IRQ_Handler_ver_4_0_begin

        SUB     LR,LR,#4          ; first of all we will save arm state
        STMDB   SP!, {LR}         ; return address and working registers
        MRS     R14, SPSR         ;
        STMDB   SP!, {R0-R12,R14}



;       ******************
;       * RESET WATCHDOG *
;       ******************

        LDR     R0,=wd_CCV        ; reset wd
        MOV     R1,#0
        STR     R1,[R0]

;         *   Used registers
;       -R0 -->
;       -R1 -->
;       -R2 -->
;       -R3 --> aic PRS
;       -R4 --> aic IRS
;       -R5 -->
;       -R6 --> Exception Vector Base addr
;       -R7 --> Vector
;       -R8 --> aic PRS addr
;       -R9 --> aic IRS addr
;       -R10 ->

        LDR     R6, =EXCEPTION_VECTOR_Base
        LDR     R8,=aic_PRS
        LDR     R9,=aic_IRS

        LDR     R4,[R9]
        LDR     R3,[R8]
        LDR     R7,[R6,R3,LSL#2]

        TEQ     R4,#0
        BEQ     IRQ_Handler_ver_4_0_end

        MOV     LR,PC                   ;branching and linking
        MOV     PC, R7

IRQ_Handler_ver_4_0_end


        LDMIA   SP!, {R0-R12,r14}
        MSR     SPSR, R14
        LDMIA   SP!, {PC}^ ; cuidadín con el ^


;******(END of ROUTINE)*****************************



;*************************************************
; SUBROUTINE INFORMATION:
;       Name:                     USER_CODE
;       Author:                   J. Granado
;       Date of creation  Dec. 21, 1998
;       Last review (4.0)  Mar. 4, 1999
;       Version                   ver 2.0
;       Comments:
;
;*************************************************
; SUBROUTINE PARAMEMTERS: NONE
;
;*************************************************
; EXECUTION CONTEXT
;
;*************************************************
USER_CODE
USER_CODE_Begin

        BL POL_DIC_00

USER_CODE_End
;********(END of ROUTINE*****************************




;*******************************************************
; SUBROUTINE INFORMATION:
;       Name:                     UART_00_IAR
```

```
;       Author:                     J. Granado
;       Date of creation   Mar 7, 1999
;       Last review (new ver)       not reviewed
;       Version                     1.0
;       Comments:
;
;
;*****************************************************
; SUBROUTINE PARAMEMTERS: NONE
;
;*****************************************************
; EXECUTION CONTEXT
;       timer can interrupt too
;
;*****************************************************
; LABELS
;
;
;*****************************************************

UART_00_IAR
UART_00_IAR_Begin

        STMDB    SP!, {LR}

UART_00_IAR_ver_3_0_Begin
        ;       R0  -> [R6], bytes counter
        ;       R1  -> received or trasmitted bytes
        ;       R2  -> [R8]
        ;       R3  -> number of received bytes
        ;       R4  -> number of transmitted bytes
        ;       R5  -> int enable reg. for next time
        ;       R6  -> aic int request status addr
        ;       R7  -> uart interrupt enable register addr
        ;       R8  -> uart interrupt identification reg
        ;       R9  -> uart rx buffer reg / tx holding reg
        ;       R11 -> flag de final de proceso. En ram.
        ;       R12 ->

        LDR     R6, =aic_IRS
        LDR     R7, =uart_IER
        LDR     R8, =uart_IIR
        LDR     R9, =uart_RBR
        LDR     R11,=EOT_addr

        ; first test aic to find uart irq source
        ; if there is no irq source active, jump to end

        LDR     R0, [R6]
        MOV     R1, #0x1                        ; test aic
        MOV     R1,R1,LSL #SER_A
        TST     R1,R0
        BEQ     UART_00_IAR_ver_3_0_End


        ; if there is uart irq source, check to find
        ; if it is rx or tx irq
        ; First rx


UART_00_IAR_ver_3_0_Label_0

        LDRB    R2, [R8]          ; read interrupt  identification regist

UART_00_IAR_ver_3_0_Label_1

        TST     R2, #0x04                       ; RX interrupt
        BEQ     UART_00_IAR_ver_3_0_End

        LDRB    R1, [R9]          ; read uart buffer
        STRB    R1,[R11]          ; store read byte into RAM

UART_00_IAR_ver_3_0_End

        LDMIA   SP!, {PC}


;******(END of ROUTINE)*****************************




;*****************************************************
; SUBROUTINE INFORMATION:
;       Name:                     TIMER_0_00_IAR
;       Author:                   J. Granado
```

```
;        Date of creation   Mar 7th, 1999
;        Last review (new ver)       not reviewed
;        Version                     1.0
;        Comments:          check in ARM debugger (Mar 8th,99)
;
;
;*************************************************
; SUBROUTINE PARAMEMTERS: NONE
;
;*************************************************
; EXECUTION CONTEXT
;        Interrupt attention routine
;
;*************************************************
; LABELS
;
;
;*************************************************

TIMER_0_00_IAR

        STMDB    SP!, {LR}

TIMER_0_00_IAR_Begin

;    Used registers
;        -R0 --> data
;        -R1 --> data
;        -R6 --> AIC interrupt request status addr.
;        ---------------------------------------------

        MOV      R0,#0x1
        MOV      R0,R0,LSL #TIMER1  ; first of all check timer2 irq source
        LDR      R6, =aic_IRS
        LDR      R1, [R0]
        TST      R1,R0
        BEQ      TIMER_0_00_IAR_End

;        ---------------------------------------------


;    Used registers
;        -R0 --> data
;        -R1 --> mask to test lsr of the uart
;        -R2 --> register
;        -R3 --> lsr
;        -R4 --> n° of bytes contained in register r2
;        -R5 --> *reg
;        -R6 --> uart rbr
;        -R7 --> **reg
;        -R8 --> *lsr
;        -R9 --> * number of registers to send
;        -R10 --> number of registers to send
;        ---------------------------------------------
        LDR      R6, =uart_RBR
        LDR      R7, =DTI_Addr+4
        LDR      R9, =DTI_Addr
        LDR      R10, [R9]
        MOV      R1,#1
        MOV      R1,R1,LSL#6

TIMER_0_00_IAR_Label_0

        LDR      R5,[R7]
        LDR      R4,[R7,#4]
        LDR      R2,[R5]

TIMER_0_00_IAR_Label_1

        LDR      R3,[R8]
        TST      R3,R1                      ; uart available ?
        BEQ      TIMER_0_00_IAR_Label_1
        STRB     R2,[R6]                    ; tx data
        MOV      R2,R2,LSR#8                ; shift 8 bits
        SUB      R4,R4,#1           ; cont--
        CMP      R4,#0x0                    ; finished register ?
        BGT      TIMER_0_00_IAR_Label_1

        SUB      R10,R10,#1                 ; any other register ?
        CMP      R10,#0x0
        BEQ      TIMER_0_00_IAR_Label_2
        ADD      R7,R7,#0x8                 ; which register ?
        B        TIMER_0_00_IAR_Label_0

;        ----------------------------
;        Clear interrupt in timer 1
;        ----------------------------
```

```
TIMER_0_00_IAR_Label_2
        LDR     R12, =timer_TC0_CI
        MOV     R0,#0
        STR     R0,[R12]

TIMER_0_00_IAR_End

        LDMIA   SP!, {PC}

;******(END of ROUTINE)*****************************



;*******************************************************
; SUBROUTINE INFORMATION:
;       Name:                   UART_00_ICR
;       Author:                 J. Granado
;       Date of creation  Mar 8th, 1999
;       Last review (new ver)   not reviewed
;       Version                 1.0
;       Comments:         based on UART_TEST_ver_2
;
;
;*************************************************
; SUBROUTINE PARAMEMTERS:
;       - R0 --> Interrupt Vector Addr
;       - R1 --> Priority
;       - R2 --> baud rate
;       - R3 --> line control register
;       - R4 --> interrup enable register
;       - R5 --> modem control
;*************************************************
; EXECUTION CONTEXT
;       Interrupt Configuration Routine
;
;*************************************************
; LABELS
;
;
;*************************************************

UART_00_ICR

        STMDB   SP!, {LR}

UART_00_Begin


        CMP     R4,#0x0
        BEQ     UART_00_ICR_Label_1

;       ---------------------------------------------
;        SET UP AIC PARAMETERS
;       ---------------------------------------------

        LDR     R10, =aic_IES               ;enable uart irq in aic
        MOV     R6, #0x1
        MOV     R6, R6, LSL  #SER_A
        STR     R6, [R10]

        LDR     R10,=EXCEPTION_VECTOR_Base
        LDR     R9,=EXCEPTION_PRIORITY_Base
        LDR     R6,=SER_A
        MOV     R6, R6, LSL #2
        STR     R1,[R9,R6]          ; priority
        STR     R0, [R10,R6]        ; vector

;       ---------------------------------------------
;       UART PARAMETERS
;       ---------------------------------------------

;        Register used:
;                - R0..5  -> reserved
;                - R6     -> DLL
;                - R7     -> IER
;                - R8     -> MCR
;                - R9     -> LCR
;                - R10     ->
;                - R11     ->

UART_00_ICR_Label_1

        LDR     R9, =uart_LCR
        LDR     R6, =uart_DLL
        LDR     R7, =uart_IER
        LDR     R8, =uart_MCR
```

```
        MOV     R1, #0x80
        STRB    R1, [R5]        ;DLAB=1

        STRB    R2, [R6]
        MOV     R2, R2,LRS#8
        STRB    R2,[R6,#1]        ; baud rate configuration
        STRB    R3,[R9]           ; line control configuration
        STRB    R4, [R7]  ; enable irq for tx and rx
        STRB    R5, [R8]  ; modem control

UART_00_ICR_End

        LDMIA   SP!, {PC}

;******(END of ROUTINE)*****************************


;*******************************************************
; SUBROUTINE INFORMATION:
;       Name:                   TIMER_0_00_ICR
;       Author:                 J. Granado
;       Date of creation  Mar 9th, 1999
;       Last review (new ver)   not reviewed
;       Version                 1.0
;       Comments:         based on UART_00_ICR
;
;
;*************************************************
; SUBROUTINE PARAMEMTERS:
;       - R0 --> Interrupt Vector Addr
;       - R1 --> Priority
;       - R2 --> load register
;       - R3 --> control register
;*************************************************
; EXECUTION CONTEXT
;       Interrupt Configuration Routine if timer 0
;
;*************************************************
; LABELS
;
;
;*************************************************

TIMER_0_00_ICR

        STMDB   SP!, {LR}

TIMER_0_00_Begin
;       ---------------------------------------------
;        SET UP AIC PARAMETERS
;       ---------------------------------------------

        LDR     R10, =aic_IES             ;enable uart irq in aic
        MOV     R6, #0x1
        MOV     R6, R6, LSL  #TIMER0
        STR     R6, [R10]

        LDR     R10,=EXCEPTION_VECTOR_Base
        LDR     R9,=EXCEPTION_PRIORITY_Base
        LDR     R6,=TIMER0
        MOV     R6, R6, LSL #2
        STR     R1,[R9,R6]        ; priority
        STR     R0, [R10,R6]      ; vector

;       ---------------------------------------------
;       set up timer paramet.
;       ---------------------------------------------
        LDR     R7,=timer_TC0_LR
        LDR     R8=timer_TC0_CR
        STR     R0,[R7]           ; load reg.
        STR     R1,[R8]           ; control reg.

TIEMR_0_00_ICR_End

        LDMIA   SP!, {PC}

;******(END of ROUTINE)*****************************

;*******************************************************
; SUBROUTINE INFORMATION:
;       Name:                   UART_00_ICR
;       Author:                 J. Granado
;       Date of creation  Mar 8th, 1999
;       Last review (new ver)   not reviewed
;       Version                 1.0
;       Comments:         based on UART_TEST_ver_2
```

```
;
;
;*****************************************************
; SUBROUTINE PARAMEMIERS:
;        - R0 --> Interrupt Vector Addr
;        - R1 --> Priority
;        - R2 --> baud rate
;        - R3 --> line control register
;        - R4 --> internup enable register
;        - R5 --> modem control
;*****************************************************
; EXECUTION CONTEXT
;        Interrupt Configuration Routine
;
;*****************************************************
; LABELS
;
;
;*****************************************************

UART_00_ICR

        STMDB    SP!, {LR}

UART_00_Begin
;       ---------------------------------------------
;        SET UP AIC PARAMETERS
;       ---------------------------------------------

        LDR      R10, =aic_IES              ;enable uart irq in aic
        MOV      R6, #0x1
        MOV      R6, R6, LSL  #SER_A
        STR      R6, [R10]

        LDR      R10,=EXCEPTION_VECTOR_Base
        LDR      R9,=EXCEPTION_PRIORITY_Base
        LDR      R6,=SER_A
        MOV      R6, R6, LSL #2
        STR      R1,[R9,R6]         ; priority
        STR      R0, [R10,R6]       ; vector

;       ---------------------------------------------
;       UART PARAMETERS
;       ---------------------------------------------

;        Register used:
;                - R0..5  -> reserved
;                - R6     -> DLL
;                - R7     -> IER
;                - R8     -> MCR
;                - R9     -> LCR
;                - R10     ->
;                - R11     ->

UART_00_ICR_Label_1

        LDR      R9, =uart_LCR
        LDR      R6, =uart_DLL
        LDR      R7, =uart_IER
        LDR      R8, =uart_MCR

        MOV      R1, #0x80
        STRB     R1, [R5]        ;DLAB=1

        STRB     R2, [R6]
        MOV      R2, R2,LRS#8
        STRB     R2,[R6,#1]         ; baud rate configuration
        STRB     R3,[R9]            ; line control configuration
        STRB     R4, [R7]  ; enable irq for tx and rx
        STRB     R5, [R8]  ; modem control

UART_00_ICR_End

        LDMIA    SP!, {PC}

;******(END of ROUTINE)*****************************


;*****************************************************
; SUBROUTINE INFORMATION:
;       Name:                      TIMER_0_00_ICR
;       Author:                    J. Granado
;       Date of creation  Mar 9th, 1999
;       Last review (new ver)      not reviewed
;       Version                    1.0
;       Comments:         based on UART_00_ICR
;
```

```
;
;*****************************************************
; SUBROUTINE PARAMEMTERS:
;        - R0 --> Interrupt Vector Addr
;        - R1 --> Priority
;        - R2 --> load register
;        - R3 --> control register
;*****************************************************
; EXECUTION CONTEXT
;        Interrupt Configuration Routine if timer 0
;
;*****************************************************
; LABELS
;
;
;*****************************************************

TIMER_0_00_ICR

        STMDB    SP!, {LR}

TIMER_0_00_Begin
;       ---------------------------------------------
;        SET UP AIC PARAMETERS
;       ---------------------------------------------

        LDR      R10, =aic_IES              ;enable uart irq in aic
        MOV      R6, #0x1
        MOV      R6, R6, LSL  #TIMER0
        STR      R6, [R10]

        LDR      R10,=EXCEPTION_VECTOR_Base
        LDR      R9,=EXCEPTION_PRIORITY_Base
        LDR      R6,=TIMER0
        MOV      R6, R6, LSL #2
        STR      R1,[R9,R6]         ; priority
        STR      R0, [R10,R6]       ; vector

;       ---------------------------------------------
;       set up timer paramet.
;       ---------------------------------------------
        LDR      R7,=timer_TC0_LR
        LDR      R8=timer_TC0_CR
        STR      R0,[R7]            ; load reg.
        STR      R1,[R8]            ; control reg.

TIEMR_0_00_ICR_End

        LDMIA    SP!, {PC}

;******(END of ROUTINE)*****************************

;*****************************************************
; SUBROUTINE INFORMATION:
;       Name:                       POL_DTC_00_main
;       Author:                     J. Granado
;       Date of creation  Mar 9, 1999
;       Last review (4.0)  ----
;       Version                     ver 1.0
;       Comments:         this routine is an
;                                   approach of a future
;                                   initialisation routine
;
;*****************************************************
; SUBROUTINE PARAMEMTERS: NONE
;
;*****************************************************
; SUBROUTINE CALLS:
;       - TIMER_0_00_ICR
;       - UART_00_ICR
;
;
;*****************************************************
; EXECUTION CONTEXT: after reset routine
;       - irq enabled:
;                (*) timer and rx_uart
;       - fiq disabled
;       - supervisor mode
;       - arm mode
;       - main program
;*****************************************************
POL_DTC_00
POL_DTC_00_Begin



;       ---------------------------------------------
```

```
;       UART congiguration (branching to UART_00_ICR:
;                - R0 <- 0
;                - R1 <- 0
;                - R2 <- baud rate 9600bps (39 = 0x27)
;                - R3 <- line control register (0x7)
;                - R4 <- interrupt enable register (0x0)
;                - R5 <- modem control reg:DTR=RTS='1'(0x3)
;       ---------------------------------------------

        MOV     R0, #0x0
        MOV     R1, #0x0
        MOV     R2, #0x27
        MOV     R3, #0x7
        MOV     R4, #0x0
        MOV     R5, #0x3
        BL      UART_00_ICR

;       ---------------------------------------------
;       TX CONTROL FRAME 'i'm alive' (branching to UART_01:
;                - R0 <- number of byte to tx
;                - R1 <- base addr
;       ---------------------------------------------
        LDR     R6,=UDA_1_Begin
        MOV     R0,#ID_2
        STRB    R0,[R6]
        LDR     R0, =IM_ALIVE
        STRB    R0, [R6,#1]

        MOV     R0,#2
        LDR     R1,=UDA_1_Begin
        BL      UART_01

;       ---------------------------------------------
;       Rx remote configuration
;       ---------------------------------------------
;       REGISTERS;
;                -R0 -> temp
;                -R1 -> temp
;                -R2 -> data received
;                -R3 -> ERROR REG.(which will be stored in LDA_1)
;                -R4 -> LSR
;                -R6 -> *uart_RBR
;                -R7 -> *UART_LSR
;                -R8 -> Uart Data Area 1, used to recieve config info

        LDR     R6,=uart_RBR
        LDR     R7, =uart_LSR
        LDR     R8, =UDA_1_Begin
        MOV     R3, 0x0

POL_DTC_00_Label_0

        LDR     R4, [R7]  ; lsr

        TST     R4, #0x1  ; DATA AVAILABLE ?
        BEQ     POL_DTC_main_Label_0

        LDRB    R2, [R6]  ; store it & check reception

;       ++++++++++++++++++
;       check reception
;       ++++++++++++++++++
        TST     R4, #0x2
        ORRNE   R3,R3,#0x2        ; overrun error has ocurred
        TST     R4, #0x4
        ORRNE   R3,R3,#0x4        ; parity error has ocurred
        TST     R4, #0x8
        ORRNE   R3,R3,#0x8        ; framing error has ocurred
        TST     R4, #0x10
        ORRNE   R3,R3,#0x10       ; break error has ocurred

;       ++++++++++++++++++
;       check frame info
;       ++++++++++++++++++
        TST     R2, #ID_1 ; Control frame
        ORRNE   R3,R3,#0x100      ; Detected frame TDF_00_1
                                  ; next byte is a control one
        BNE     POL_DTC_main_Label_1

        TST     R2, #ID_2 ; Modem configuration frame
        ORRNE   R3,R3,#0x200      ; Detected frame TDF_00_2
        BNE     POL_DTC_main_Label_0
        TST     R2, #ID_3 ; TIMER configuration
        ORRNE   R3,R3,#0x400      ; Detected frame TDF_00_3
        BNE     POL_DTC_main_Label_0
        TST     R2, #ID_4 ; DTI
        ORRNE   R3,R3,#0x800      ; Detected frame TDF_00_4
```

```
        BNE     POL_DTC_main_Label_0

        STRB    R2,[R8],#1          ; store data if it is not an ID

POL_DTC_main_Label_1

        LDR     R4, [R7] ; lsr
        TST     R4, #0x1 ; DATA AVAILABLE ?
        BEQ     POL_DTC_main_Label_1
        LDRB    R2, [R6] ;

        TST     R2,#EOC
        BNE     POL_DTC_main_Label_0

        LDR     R6,=LDA_1_Begin
        STR     R3,[R6]

;       ++++++++++++++++++
;       Config device SBM according to
;       the data sent from the PC
;       ++++++++++++++++++

        LDR     R6,=sbm_IER
        LDR     R7,=sbm_CCR
        LDR     R8,=sbm_BDT
        LDR     R9,=sbm_BPT
        LDR     R10,=sbm_FREQ

        LDR     R5,=UDA_1_Begin
        LDRB    R0,[R5]
        STR     R0,[R6]

        LDRB    R0,[R6,#1]
        LDRB    R1,[R6,#2]
        MOV     R1,R1,LSL#8
        ORR     R0,R0,R1
        STR     R0,[R7]

        LDRB    R0,[R6,#3]
        LDRB    R1,[R6,#4]
        MOV     R1,R1,LSL#8
        ORR     R0,R0,R1
        STR     R0,[R8]

        LDRB    R0,[R6,#5]
        LDRB    R1,[R6,#6]
        MOV     R1,R1,LSL#8
        ORR     R0,R0,R1
        STR     R0,[R9]

        LDRB    R0,[R6,#7]
        LDRB    R1,[R6,#8]
        MOV     R1,R1,LSL#8
        ORR     R0,R0,R1
        STR     R0,[R10]



;       ++++++++++++++++++
;       Config device TIMER 0 according to
;       the data sent from the PC
;       ++++++++++++++++++
;       ------------------------------------------
;       TIMER 0 congiguration (branching to TIMER_00_ICR:
;                - R0 <- TIMER_0_00_IAR
;                - R1 <- 0
;                - R2 <- LOAD REG from { UAD[1] | UAD[0] }
;                - R3 <- control reg
;       ------------------------------------------

        LDR     R0, =TIMER_0_00_IAR
        MOV     R1, #0x0
        LDR     R5,=UDA_1_Begin
        LDRB    R2,[R5,#0x9]
        LDRB    R4,[R5,#0xa]
        MOV     R4,R4,LSL#8
        ORR     R2,R2,R4
        LDRB    R3,[R5,#0xB]
        BL      TIMER_0_00_ICR

;       ------------------------------------------
;       UART congiguration (branching to UART_00_ICR):
;                - R0 <- UART_00_IAR
;                - R1 <- 1
;                - R2 <- baud rate 125kbps (3 = 0x3)
;                - R3 <- line control register (0x7)
;                - R4 <- interrupt enable register: RX IRQ (0x01)
```

```
;                - R5 <- modem control reg:DTR=RTS='1'(0x3)
;       ------------------------------------------

        LDR     R0, =UART_00_IAR
        MOV     R1, #0x1
        MOV     R2, #0x3
        MOV     R3, #0x7
        MOV     R4, #0x1
        MOV     R5, #0x3
        BL      UART_00_ICR

;       ------------------------------------------
;       TX CONTROL FRAME 'starting simulation' (branching to UART_01:
;                - R0 <- number of byte to tx
;                - R1 <- base addr
;       ------------------------------------------
        LDR     R6,=UDA_2_Begin
        LDR     R7,=LDA_1_Begin
        MOV     R0,#ID_1
        STRB    R0,[R6]
        LDR     R0, =STARTING
        STRB    R0, [R6,#1]
        LDR     R0,[R7]
        STRB    R0, [R6,#2]
        MOV     R0,R0,LSR#8
        STRB    R0, [R6,#3]
        MOV     R0,R0,LSR#8
        STRB    R0, [R6,#4]
        MOV     R0,R0,LSR#8
        STRB    R0, [R6,#5]
        MOV     R0,#6
        LDR     R1,=UDA_1_Begin
        BL      UART_01


;       ------------------------------------------
;       Enable irq in ARM processor
;       ------------------------------------------
        MOV     R0, #Mode_SVC               ; Enable IRQ in arm
        MSR     CPSR, R0

;       ------------------------------------------
;       Start reception process
;       ------------------------------------------


;   Used registers
;       -R0 --> temp
;       -R1 --> cont
;       -R2 --> IIR
;       -R3 --> *LDA_1
;       -R4 --> *LDA_2
;       -R5 --> *LDA_3
;       -R6 --> *SBM CCR
;       -R7 --> *IIR
;       -R8 --> *RBR_0
;       -R9 --> *RBR_1
;       -R10 --> UDA_3_Begin
;       ------------------------------------------

        MOV     R1,#0x0
        MOV     R2,#0x0
        LDR     R3, =LDA_1_Begin
        LDR     R4, =LDA_2_Begin
        LDR     R5, =LDA_3_Begin
        LDR     R6, =smb_CCR
        LDR     R7, =smb_IIR
        LDR     R8, =smb_RBR_0
        LDR     R9, =smb_RBR_1
        LDR     R10, =UDA_3_Begin

        LDR     R0, [R6]  ; enable both channels
        ORR     R0,R0,#0x3
        STR     R0,[R6]

PDL_DTC_00_Label_2
        LDR     R2,[R7]
        TST     R2,#0x7           ; Any data available in channel 1 ?
        BEQ     PDL_DTC_00_Label_3
        LDR     R0,[R9]
        STRB    R0,[R4],#1
        LDR     R0,[R3]
        ADD     R0,R0,#1  ; cont ++
        STR     R0,[R3]

PDL_DTC_00_Label_3
        TST     R2,#0x38  ; Any data available in channel 0 ?
```

```
        BEQ     PDL_DTC_00_Label_4
        LDR     R0,[R8]
        STRB    R0,[R5],#1
        LDR     R0,[R3,#4]
        ADD     R0,R0,#1  ; cont ++
        STR     R0,[R3,#4]

PDL_DTC_00_Label_4
        LDRB    R0,[R10]  ; End of sim ?
        TST     R0,#EOS
        BEQ     PDL_DTC_00_Label_2

;       ---------------------------------------------
;       Send to PC the info recieved
;       ---------------------------------------------


;       ---------------------------------------------
;       TX DATA FROM SBM  (branching to UART_01:
;                - R0 <- number of bytes to tx
;                - R1 <- base addr
;       ---------------------------------------------
        LDR     R6,=LDA_2_Begin
        LDR     R7,=LDA_3_Begin
        LDR     R8,=LDA_1_Begin
        LDR     R9,=UDA_2_Begin

        MOV     R0,#ID_4
        STRB    R0,[R6]-#1
        LDR     R0,[R8]           ; canal 1
        MOV     R1,R6
        BL      UART_01

        MOV     R0,#ID_5  ; canal 0
        STRB    R0,[R7]-#1
        LDR     R0,[R8,#4]
        MOV     R1,R7
        BL      UART_01

        MOV     R0,#ID_1  ; EOT
        STRB    R0,[R9]
        MOV     R0,#EOT
        STRB    R0,[R9,#1]
        MOV     R0,#2
        MOV     R1,R9
        BL      UART_01

POL_DTC_00_End

        END




;********(END of ROUTINE*****************************
```