XEMICS

AN8000.01

Application note

XE8000 address pack

For further information please contact
XEMICS SA,
Tel: +41 32 7205 511 *Fax: +41 32 7205 770*
www.xemics.ch

1 Introduction

This application note concerns all XE8000 products.

A standardized peripheral address naming (address pack) is provided to help developers. A copy of the Address Pack is listed below. The current version is available on XEMICS web site (http://www.xemics.ch). Further information about the use of each register can be found in the XE8000 Series data book. Further information about the programming syntax can be found in the CoolRide development tools and in the application note AN8000.04.

They are two files in this address pack. The “crt0.ld” is a linker file used for any program. The C-file is only used for programs with C-code.

2 Registers list

Left column include register name and address.
Right columns include bit name, access (r: read, r0: always 0 when read, w: write, c: cleared by writing any value, c1: cleared by writing 1), and reset status (0 or 1) and conditions (por: power-on-reset, cold: cold reset, pad: reset pad, pconf: port configuration, system: system reset, synch: synchro reset). Empty bits are reserved for future use and should not be written, neither should their read value used for any purpose as it may change without notice.

2.1 Low power RAM

Low power RAM is a small additionnal RAM area with extremely low power requirement.

Name Address	7	6	5	4	3	2	1	0
h0000	rw	rw	rw	rw	rw	rw	rw	rw
h0001	rw	rw	rw	rw	rw	rw	rw	rw
h0002	rw	rw	rw	rw	rw	rw	rw	rw
h0003	rw	rw	rw	rw	rw	rw	rw	rw
h0004	rw	rw	rw	rw	rw	rw	rw	rw
h0005	rw	rw	rw	rw	rw	rw	rw	rw
h0006	rw	rw	rw	rw	rw	rw	rw	rw
h0007	rw	rw	rw	rw	rw	rw	rw	rw

Table 2.1: Low power RAM

2.2 System, oscillators, prescaler and watchdog

Name Address	7	6	5	4	3	2	1	0
RegSysCtrl h0010	SleepEn rw, 0 por	EnRes-PConf rw, 0 cold	EnBus-Error rw, 0 cold	EnResWD rw, 0 cold				

Table 2.2: System control registers

Name Address	7	6	5	4	3	2	1	0
RegSysReset h0011	Sleep w, 0 (por, pad or cold)	ResPor r, 0	ResBus-Error rc, 0 cold	ResWD rc, 0 cold	ResPortA rc, 0 cold	ResPad-Deb rc, 0 cold	ResPad rc, 0 cold	
RegSysClock h0012	CpuSel rw, 0 sleep	ExtClk r, 0 cold	EnExtClk rw, 0 cold	BiasRC rw, 1 cold	ColdXtal r, 1 sleep	ColdRC r, 1 sleep	EnableXtal rw, 0 sleep	EnableRC rw, 1 sleep
RegSysMisc h0013	DisResPad rw				RCOnPA0 rw, 0 sleep	DebFast rw, 0 sleep	Output-CkXtal rw, 0 sleep	Output-CkCPU rw, 0 sleep
RegSysWD h0014					WatchDog(3) special	WatchDog(2) special	WatchDog(1) special	WatchDog(0) special
RegSysPre0 h0015								ResPre W, 0 cold
RegSysPre1 h0016								
RegSysTest1 h0017								
RegSysTest2 h0018								
RegSysTest3 h0019								
RegSysTestAna h001A								
RegSysRCTrim1 h001B			EnRCRExt rw, 0 cold	RCFreq-Range rw, 0 cold	RCFreq-Coarse(3) rw, 0 cold	RCFreq-Coarse(2) rw, 0 cold	RCFreq-Coarse(1) rw, 0 cold	RCFreq-Coarse(0) rw, 0 cold
RegSysRCTrim2 h001C			RCFreq-Fine(5) rw, 0 cold	RCFreq-Fine(4) rw, 0 cold	RCFreq-Fine(3) rw, 0 cold	RCFreq-Fine(2) rw, 0 cold	RCFreq-Fine(1) rw, 0 cold	RCFreq-Fine(0) rw, 0 cold
RegSysVIReg h001D								
reserved h001E								
RegSysWarm h001F								

Table 2.2: System control registers

2.3 PortA

Name Address	7	6	5	4	3	2	1	0
RegPAIn h0020	PAIn(7) r	RegPAIn(6) r	PAIn(5) r	PAIn(4) r	PAIn(3) r	PAIn(2) r	PAIn(1) r	PAIn(0) r
RegPADebounce h0021	PADeb(7) rw, 0 pconf	PADeb(6) rw, 0 pconf	PADeb(5) rw, 0 pconf	PADeb(4) rw, 0 pconf	PADeb(3) rw, 0 pconf	PADeb(2) rw, 0 pconf	PADeb(1) rw, 0 pconf	PADeb(0) rw, 0 pconf
RegPAEdge h0022	PAEdge(7) rw, 0 system	PAEdge(6) rw, 0 system	PAEdge(5) rw, 0 system	PAEdge(4) rw, 0 system	PAEdge(3) rw, 0 system	PAEdge(2) rw, 0 system	PAEdge(1) rw, 0 system	PAEdge(0) rw, 0 system
RegPAPullUp h0023	PAPullUp(7) rw, 0 pconf	PAPullUp(6) rw, 0 pconf	PAPullUp(5) rw, 0 pconf	PAPullUp(4) rw, 0 pconf	PAPullUp(3) rw, 0 pconf	PAPullUp(2) rw, 0 pconf	PAPullUp(1) rw, 0 pconf	PAPullUp(0) rw, 0 pconf
RegPARes0 h0024	PARes0(7) rw, 0 synch	PARes0(6) rw, 0 synch	PARes0(5) rw, 0 synch	PARes0(4) rw, 0 synch	PARes0(3) rw, 0 synch	PARes0(2) rw, 0 synch	PARes0(1) rw, 0 synch	PARes0(0) rw, 0 synch
RegPARes1 h0025	PARes1(7) rw, 0 synch	PARes1(6) rw, 0 synch	PARes1(5) rw, 0 synch	PARes1(4) rw, 0 synch	PARes1(3) rw, 0 synch	PARes1(2) rw, 0 synch	PARes1(1) rw, 0 synch	PARes1(0) rw, 0 synch

Table 2.3: Port A registers

Name Address	7	6	5	4	3	2	1	0
reserved h0026								
RegPATest h0027								

Table 2.3: Port A registers

2.4 PortB

Name Address	7	6	5	4	3	2	1	0
RegPBOut h0028	PBOut(7) rw, 0 pconf	PBOut(6) rw, 0 pconf	PBOut(5) rw, 0 pconf	PBOut(4) rw, 0 pconf	PBOut(3) rw, 0 pconf	PBOut(2) rw, 0 pconf	PBOut(1) rw, 0 pconf	PBOut(0) rw, 0 pconf
RegPBIn h0029	PBIn(7) r	PBIn(6) r	PBIn(5) r	PBIn(4) r	PBIn(3) r	PBIn(2) r	PBIn(1) r	PBIn(0) r
RegPBDir h002A	PBDir(7) rw, 0 pconf	PBDir(6) rw, 0 pconf	PBDir(5) rw, 0 pconf	PBDir(4) rw, 0 pconf	PBDir(3) rw, 0 pconf	PBDir(2) rw, 0 pconf	PBDir(1) rw, 0 pconf	PBDir(0) rw, 0 pconf
RegPBOpen h002B	PBOpen(7) rw, 0 pconf	PBOpen(6) rw, 0 pconf	PBOpen(5) rw, 0 pconf	PBOpen(4) rw, 0 pconf	PBOpen(3) rw, 0 pconf	PBOpen(2) rw, 0 pconf	PBOpen(1) rw, 0 pconf	PBOpen(0) rw, 0 pconf
RegPBPullUp h002C	PBPullUp(7) rw, 0 pconf	PBPullUp(6) rw, 0 pconf	PBPullUp(5) rw, 0 pconf	PBPullUp(4) rw, 0 pconf	PBPullUp(3) rw, 0 pconf	PBPullUp(2) rw, 0 pconf	PBPullUp(1) rw, 0 pconf	PBPullUp(0) rw, 0 pconf
RegPBAna h002D					PBAna(3) rw, 0 pconf	PBAna(2) rw, 0 pconf	PBAna(1) rw, 0 pconf	PBAna(0) rw, 0 pconf
reserved h002E								
reserved h002F								

Table 2.4: Port B registers

2.5 PortC

Name Address	7	6	5	4	3	2	1	0
RegPCOut h0030	PCOut(7) rw, 0 pconf	PCOut(6) rw, 0 pconf	PCOut(5) rw, 0 pconf	PCOut(4) rw, 0 pconf	PCOut(3) rw, 0 pconf	PCOut(2) rw, 0 pconf	PCOut(1) rw, 0 pconf	PCOut(0) rw, 0 pconf
RegPCIn h0031	PCIn(7) r	PCIn(6) r	PCIn(5) r	PCIn(4) r	PCIn(3) r	PCIn(2) r	PCIn(1) r	PCIn(0) r
RegPCDir h0032	PCDir(7) rw, 0 pconf	PCDir(6) rw, 0 pconf	PCDir) rw, 0 pconf	PCDir(4) rw, 0 pconf	PCDir(3) rw, 0 pconf	PCDir(2) rw, 0 pconf	PCDir(1) rw, 0 pconf	PCDir(0) rw, 0 pconf
reserved h0033								

Table 2.5: Port C registers

2.6 MTP

Name Address	7	6	5	4	3	2	1	0
RegEEP h0038	rw	rw	rw	rw	rw	rw	rw	rw
RegEEP1 h0039	rw	rw	rw	rw	rw	rw	rw	rw

Table 2.6: MTP control registers

Name Address	7	6	5	4	3	2	1	0
RegEEP2 h003A	special	special	special	special	special	special	special	special
RegEEP3 h003B	special	special	special	special	special	special	special	special

Table 2.6: MTP control registers

2.7 Events

Name Address	7	6	5	4	3	2	1	0
RegEvn h003C	EvnCntA rc1, 0 system	EvnCntC rc1, 0 system	EvnPre1 rc1, 0 system	EvnPA(1) rc1, 0 system	EvnCntB rc1, 0 system	EvnCntD rc1, 0 system	EvnPre2 rc1, 0 system	EvnPA(0) rc1, 0 system
RegEvnEn h003D	EvnEnCntA rw, 0 system	EvnEnCntC rw, 0 system	EvnEnPre1 rw, 0 system	EvnEnPA(1) rw, 0 system	EvnEnCntB rw, 0 system	EvnEnCntD rw, 0 system	EvnEnPre2 rw, 0 system	EvnEnPA(0) rw, 0 system
RegEvnPriority h003E	EvnPriority(7) r,1 system	EvnPriority(6) r,1 system	EvnPriority(5) r,1 system	EvnPriority(4) r,1 system	EvnPriority(3) r,1 system	EvnPriority(2) r,1 system	EvnPriority(1) r,1 system	EvnPriority(0) r,1 system
RegEvnEvn h003F							EvnHigh r, 0 system	EvnLow r, 0 system

Table 2.7: Events control registers

2.8 Interrupts

Name Address	7	6	5	4	3	2	1	0
RegIrqHig h0040	IrqAc rc1, 0 system	IrqPre1 rc1, 0 system		IrqCntA rc1, 0 system	IrqCntC rc1, 0 system		IrqUartTx rc1, 0 system	IrqUartRx rc1, 0 system
RegIrqMid h0041			IrqPA(5) rc1, 0 system	IrqPA(4) rc1, 0 system	IrqPre2 rc1, 0 system	IrqVld rc1, 0 system	IrqPA(1) rc1, 0 system	IrqPA(0) rc1, 0 system
RegIrqLow h0042	IrqPA(7) rc1, 0 system	IrqPA(6) rc1, 0 system	IrqCntB rc1, 0 system	IrqCntD rc1, 0 system	IrqPA(3) rc1, 0 system	IrqPA(2) rc1, 0 system		
RegIrqEnHig h0043	IrqEnAc rw, 0 system	IrqEnPre1 rw, 0 system		IrqEnCntA rw, 0 system	IrqEnCntC rw, 0 system		IrqEnUartTx rw, 0 system	IrqEnUartRx rw, 0 system
RegIrqEnMid h0044			IrqEnPA(5) rw, 0 system	IrqEnPA(4) rw, 0 system	IrqEnPre2 rw, 0 system	IrqEnVld rw, 0 system	IrqEnPA(1) rw, 0 system	IrqEnPA(0) rw, 0 system
RegIrqEnLow h0045	IrqEnPA(7) rw, 0 system	IrqEnPA(6) rw, 0 system	IrqEnCntB rw, 0 system	IrqEnCntD rw, 0 system	IrqEnPA(3) rw, 0 system	IrqEnPA(2) rw, 0 system		
RegIrqPriority h0046	IrqPriority(7) r, 1 system	IrqPriority(6) r, 1 system	IrqPriority(5) r, 1 system	IrqPriority(4) r, 1 system	IrqPriority(3) r, 1 system	IrqPriority(2) r, 1 system	IrqPriority(1) r, 1 system	IrqPriority(0) r, 1 system
RegIrqIrq h0047						IrqHig r, 0 system	IrqMid r, 0 system	IrqLow r, 0 system

Table 2.8: Interrupts control registers

2.9 USRT

Name Address	7	6	5	4	3	2	1	0
RegUsrtSin h0048								UsrtSin rw, 1 system
RegUsrtScl h0049								UsrtScl rw, 1 system

Table 2.9: USRT control registers

Name Address	7	6	5	4	3	2	1	0
RegUsrtCtrl h004A						rw, must be 0	rw, must be 0	UsrtEnable rw, 0 system
reserved h004B								
reserved h004C								
RegUsrtData h004D								UsrtData r
RegUsrtEdgeScl h004E								UsrtEdgeScl r, 0 system
reserved h004F								

Table 2.9: USRT control registers

2.10 UART

Name Address	7	6	5	4	3	2	1	0
RegUartCtrl h0050	UartEcho rw, 0 system	UartEnRx rw, 0 system	UartEnTx rw, 0 system	UartXRx rw, 0 system	UartXTx rw, 0 system	UartBR(2) rw, 1 system	UartBR(1) rw, 0 system	UartBR(0) rw, 1 system
RegUartCmd h0051	SelXtal rw, 0 system	UartWakeup rw, 0 system	UartRCSel(2) rw, 0 system	UartRCSel(1) rw, 0 system	UartRCSel(0) rw, 0 system	UartPM rw, 0 system	UartPE rw, 0 system	UartWL rw, 1 system
RegUartTx h0052	UartTx(7) rw, 0 system	UartTx(6) rw, 0 system	UartTx(5) rw, 0 system	UartTx(4) rw, 0 system	UartTx(3) rw, 0 system	UartTx(2) rw, 0 system	UartTx(1) rw, 0 system	UartTx(0) rw, 0 system
RegUartTxSta h0053							UartTxBusy r, 0 system	UartTxFull r, 0 system
RegUartRx h0054	UartRx(7) r	UartRx(6) r	UartRx(5) r	UartRx(4) r	UartRx(3) r	UartRx(2) r	UartRx(1) r	UartRx(0) r
RegUartRxSta h0055			UartRxSErr r	UartRxPErr r	UartRxFErr r	UartRxOErr c	UartRxBusy r	UartRxFull r
reserved h0056								
reserved h0057								

Table 2.10: UART control registers

2.11 Counters

Name Address	7	6	5	4	3	2	1	0
RegCntA h0058	CounterA(7) rw	CounterA(6) rw	CounterA(5) rw	CounterA(4) rw	CounterA(3) rw	CounterA(2) rw	CounterA(1) rw	CounterA(0) rw
RegCntB h0059	CounterB(7) rw	CounterB(6) rw	CounterB(5) rw	CounterB(4) rw	CounterB(3) rw	CounterB(2) rw	CounterB(1) rw	CounterB(0) rw
RegCntC h005A	CounterC(7) rw	CounterC(6) rw	CounterC(5) rw	CounterC(4) rw	CounterC(3) rw	CounterC(2) rw	CounterC(1) rw	CounterC(0) rw
RegCntD h005B	CounterD(7) rw	CounterD(6) rw	CounterD(5) rw	CounterD(4) rw	CounterD(3) rw	CounterD(2) rw	CounterD(1) rw	CounterD(0) rw
RegCntCtrlCk h005C	CntDSel(1) rw	CntDSel(0) rw	CntCSel(1) rw	CntCSel(0) rw	CntBSel(1) rw	CntBSel(0) rw	CntASel(1) rw	CntASel(0) rw

Table 2.11: Counters control registers

Name Address	7	6	5	4	3	2	1	0
RegCntConfig1 h005D	CntDDownUp rw	CntCDownUp rw	CntBDownUp rw	CntADownUp rw	CascadeCD rw	CascadeAB rw	CntPWM1 rw, 0 system	CntPWM0 rw, 0 system
RegCntConfig2 h005E	CapSel(1) rw, 0 system	CapSel(0) rw, 0 system	CapFunc(1) rw, 0 system	CapFunc(0) rw, 0 system	PWM1Size(1) rw	PWM1Size(0) rw	PWM0Size(1) rw	PWM0Size(0) rw
RegCntOn h005F					CntDEnable rw, 0 system	CntCEnable rw, 0 system	CntBEnable rw, 0 system	CntAEnable rw, 0 system

Table 2.11: Counters control registers

2.12 Acquisition chain

Name Address	7	6	5	4	3	2	1	0
RegACOutLSB h0060	ADCOutL(7) r	ADCOutL(6) r	ADCOutL(5) r	ADCOutL(4) r	ADCOutL(3) r	ADCOutL(2) r	ADCOutL(1) r	ADCOutL(0) r
RegACOutMSB h0061	ADCOutM(7) r	ADCOutM(6) r	ADCOutM(5) r	ADCOutM(4) r	ADCOutM(3) r	ADCOutM(2) r	ADCOutM(1) r	ADCOutM(0) r
RegACCfg0 h0062	Start r0w, 0 system	NelConv(1) rw, 0 system	NelConv(0) rw, 1 system	OSR(2) rw, 0 system	OSR(1) rw, 1 system	OSR(0) rw, 0 system	Cont rw, 0 system	
RegACCfg1 h0063	IbAmpADC(1) rw, 1 system	IbAmpAdc(0) rw, 1 system	IbAmpPga(1) rw, 1 system	IbAmpPga(0) rw, 1 system	Enable(3) rw, 0 system	Enable(2) rw, 0 system	Enable(1) rw, 0 system	Enable(0) rw, 1 system
RegACCfg2 h0064	Fin(1) rw, 0 system	Fin(0) rw, 0 system	Pga2Gain(1) rw, 0 system	Pga2Gain(0) rw, 0 system	Pga2Off(3) rw, 0 system	Pga2Off(2) rw, 0 system	Pga2Off(1) rw, 0 system	Pga2Off(0) rw, 0 system
RegACCfg3 h0065	Pga1Gain rw, 0 system	Pga3Gain(6) rw, 0 system	Pga3Gain(5) rw, 0 system	Pga3Gain(4) rw, 0 system	Pga3Gain(3) rw, 1 system	Pga3Gain(2) rw, 1 system	Pga3Gain(1) rw, 0 system	Pga3Gain(0) rw, 0 system
RegACCfg4 h0066		Pga3Off(6) rw, 0 system	Pga3Off(5) rw, 0 system	Pga3Off(4) rw, 0 system	Pga3Off(3) rw, 0 system	Pga3Off(2) rw, 0 system	Pga3Off(1) rw, 0 system	Pga3Off(0) rw, 0 system
RegACCfg5 h0067	Busy r, 0 system	Def wr0	AMux(4) rw, 0 system	AMux(3) rw, 0 system	AMux(2) rw, 0 system	AMux(1) rw, 0 system	AMux(0) rw, 0 system	VMux rw, 0 system

Table 2.12: Acquisition chain control registers

2.13 DACs

Name Address	7	6	5	4	3	2	1	0
RegDASInLsb h0074	DASInLSB(7) w	DASInLSB(6) w	DASInLSB(5) w	DASInLSB(4) w	DASInLSB(3) w	DASInLSB(2) w	DASInLSB(1) w	DASInLSB(0) w
RegDASInMsb h0075	DASInMSB(7) w	DASInMSB(6) w	DASInMSB(5) w	DASInMSB(4) w	DASInMSB(3) w	DASInMSB(2) w	DASInMSB(1) w	DASInMSB(0) w
RegDASCfg0 h0076	NSOrder(1) rw, 0 system	NSOrder(0) rw, 0 system	CodeIMax(2) rw, 0 system	CodeIMax(1) rw, 0 system	CodeIMax(0) rw, 0 system	DASEnable(1) rw, 0 system	DASEnable(0) rw, 0 system	Fin rw, 0 system
RegDASCfg1 h0077							BW rw, 0 system	Inv rw, 0 system
RegDAB1In h0078	DABIn(7) w	DABIn(6) w	DABIn(5) w	DABIn(4) w	DABIn(3) w	DABIn(2) w	DABIn(1) w	DABIn(0) w
RegDAB1Cfg h0079							DAB1- Enable(1) rw, 0 system	DAB1- Enable(0) rw, 0 system
reserved h007A								
reserved h007B								

Table 2.13: DACs control registers

2.14 Vmult and Vld registers

Name Address	7	6	5	4	3	2	1	0
RegVmultCfg1 h007C								
RegVmultCfg0 h007D						Enable rw, 0 system	Fin(1) rw, 0 system	Fin(0) rw, 0 system
RegVldCtrl h007E					VlfMult rw, 0 cold	VldTune(2) rw, 0 cold	VldTune(1) rw, 0 cold	VldTune(0) rw, 0 cold
RegVldStat h007F						VldIrq r, 0 system	VldValid r, 0 system	VldEn rw, 0 system

Table 2.14: Vmult and Vld control registers

3 Listing of the address pack

3.1 Register names and addresses declaration for the linker

This file is used with CoolRide development tools. Any assembler file can use the variables defined below. Their addresses will be set by the linker when building the code. This file is named “crt0.ld” and has to be placed in the link file folder.

Due to the organization of the GNU tools that are used by CoolRide, the generated code includes the program with a virtual non-zero start address. When placed in ROM or MTP this will be mapped to normal program address space. For the same reason, instructions all use 4 bytes and are mapped to the 22 bits wide program memory when transferred to the final ROM/MTP.

```
/*********************************************************************
**
** crt0.ld:  CoolRisc 816 linker script
**
** Author: Xemics SA
**
** Modifications:
**
** 20-05-98/PMr: General review
** 29-10-98/MiL: crt0.ld
**
*********************************************************************/

/*********************************************************************
**
** Memory area definition
**
** prog_rom      : program memory space
** data_io_ext   : external data IO space
** data_ram_int0 : internal data RAM page 0 space
** data_ram_int  : internal data RAM space
**
**
** Sections definition
**
** .text         : program space, ELF’s 0x100000..0x13ffff addressing range
** .bss          : data without initialization value
** .page0_bss    : page 0 data without initialization value
**
*********************************************************************/
```

```
MEMORY
{
  prog_rom      : ORIGIN = 0x00100000, LENGTH = 32K
    /* Each instruction is placed on 4 bytes by the compiler,
       32K / 4 = 8 kinstruction of XE8000 memory */
  data_io_ext   : ORIGIN = 0x00000000, LENGTH = 128
    /* Data registers and peripherals */
  data_ram_int0 : ORIGIN = 0x00000080, LENGTH = 128
    /* RAM on page 0 */
  data_ram_int  : ORIGIN = 0x00000100, LENGTH = 384
    /* RAM pages 1 and above */
  error_no_drom : ORIGIN = 0x00010000, LENGTH = 0
    /* No variable value assignement allowed in ROM space */
}

/************************************************************************
**
** Peripherial registers address definition.
**
************************************************************************/
Reg00                   = 0x00;
Reg01                   = 0x01;
Reg02                   = 0x02;
Reg03                   = 0x03;
Reg04                   = 0x04;
Reg05                   = 0x05;
Reg06                   = 0x06;
Reg07                   = 0x07;
/* Reg08                = 0x08; not yet available */
/* Reg09                = 0x09; not yet available */
/* Reg10                = 0x0A; not yet available */
/* Reg11                = 0x0B; not yet available */
/* Reg12                = 0x0C; not yet available */
/* Reg13                = 0x0D; not yet available */
/* Reg14                = 0x0E; not yet available */
/* Reg15                = 0x0F; not yet available */

RegSysCtrl              = 0x10;
RegSysReset             = 0x11;
RegSysClock             = 0x12;
RegSysMisc              = 0x13;
RegSysWd                = 0x14;
RegSysPre0              = 0x15;
RegSysPre1              = 0x16;
RegSysTest1             = 0x17;
RegSysTest2             = 0x18;
RegSysTest3             = 0x19;
RegSysTestAna           = 0x1A;
RegSysRcTrim1           = 0x1B;
RegSysRcTrim2           = 0x1C;
RegSysVIreg             = 0x1D;
RegSysWarm              = 0x1F;

RegPAIn                 = 0x20;
RegPADebounce           = 0x21;
RegPAEdge               = 0x22;
RegPAPullup             = 0x23;
RegPARes0               = 0x24;
RegPARes1               = 0x25;
RegPAEvent              = 0x26;
```

```
RegPATest               = 0x27;

RegPBOut                = 0x28;
RegPBIn                 = 0x29;
RegPBDir                = 0x2A;
RegPBOpen               = 0x2B;
RegPBPullup             = 0x2C;
RegPBAna                = 0x2D;
RegPBTest               = 0x2E;

RegPCOut                = 0x30;
RegPCIn                 = 0x31;
RegPCDir                = 0x32;
RegPCTest               = 0x33;

RegEEP                  = 0x38;
RegEEP1                 = 0x39;
RegEEP2                 = 0x3A;
RegEEP3                 = 0x3B;

RegEvn                  = 0x3C;
RegEvnEn                = 0x3D;
RegEvnPriority          = 0x3E;
RegEvnEvn               = 0x3F;

RegIrqHig               = 0x40;
RegIrqMid               = 0x41;
RegIrqLow               = 0x42;
RegIrqEnHig             = 0x43;
RegIrqEnMid             = 0x44;
RegIrqEnLow             = 0x45;
RegIrqPriority          = 0x46;
RegIrqIrq               = 0x47;

RegUartCtrl             = 0x50;
RegUartCmd              = 0x51;
RegUartTx               = 0x52;
RegUartTxSta            = 0x53;
RegUartRx               = 0x54;
RegUartRxSta            = 0x55;
RegUartTest             = 0x56;

RegCntA                 = 0x58;
RegCntB                 = 0x59;
RegCntC                 = 0x5A;
RegCntD                 = 0x5B;
RegCntCtrlCk            = 0x5C;
RegCntConfig1           = 0x5D;
RegCntConfig2           = 0x5E;
RegCntOn                = 0x5F;

RegAcOutLsb             = 0x60;
RegAcOutMsb             = 0x61;
RegAcCfg0               = 0x62;
RegAcCfg1               = 0x63;
RegAcCfg2               = 0x64;
RegAcCfg3               = 0x65;
RegAcCfg4               = 0x66;
RegAcCfg5               = 0x67;

RegDasInLsb             = 0x74;
```

```
RegDasInMsb             = 0x75;
RegDasCfg0              = 0x76;
RegDasCfg1              = 0x77;

RegDab1In               = 0x78;
RegDab1Cfg              = 0x79;

RegVmultCfg0            = 0x7C;
RegVmultCfg1            = 0x7D;

RegVldCtrl              = 0x7E;
RegVldStat              = 0x7F;


/*****************************************************************************
**
** Relocation of the top of the stack
**
*****************************************************************************/
_estack = 0x0200;

/*****************************************************************************
**
** Stick sections into the various memory area, GNU tools syntax
**
*****************************************************************************/
SECTIONS
{
      /* the program code is stored in the .text section */
      .text :
      {
            _stext = .;
            *(.text )
            _etext = .;
            _eprom = .;
      } > prog_rom


      /* the uninitialized variables are stored into the .bss section */
      .bss (NOLOAD) :
      {
            _sbss = .;
            *(.bss)
            *(COMMON)
            _ebss = .;
            /* Location of the _heapBottom pointer address, if needed */
            _heapBottom = .;
      } > data_ram_int  /*data_ram_ext*/

      .page0_bss (NOLOAD) :
      {
            _spage0bss = .;
            *(.page0_bss)
            _epage0bss = .;
      } > data_ram_int0

      .page0_data  :  {
            _spage0data = .;
            *(.page0_data)
            _epage0data = .;
      } > error_no_drom
```

```
      .data  :  {
            _sdata = .;
            *(.data)
            _edata = .;
      } > error_no_drom

      .rodata  :  {
            _srodata = .;
            *(.rodata)
            _rodata = .;
      } > error_no_drom

      /* these sections contain debug information */
      .stab :
      {
            *(.stab)
      }

      .stabstr :
      {
            *(.stabstr)
      }
}
```

3.2 Registers names and address declaration for the C compiler

The file listed below should be referenced by any C file using system or peripheral registers. It requires the “crt0.ld” file in the linker (see above).

The way this file is built, register names are not seen in the debugger where they should be observed using their address (monitor address 0x02 to see Reg02). The link between names and addresses is made in the “crt0.ld” file (see above).

As data and program memory are separated on the circuit (the CPU can only execute instructions, it can not read it on its data bus), one should not declare variables with a preset value (table of constants, etc...), but declare empty variables and then assign values to the variables (the compiler will first reserve the necessary address space and then generate a MOVE instruction for each value).

```
//*****************************************************************
// File       : XE8000.h
//*****************************************************************
//
// Version    : V 1.0
//
// Written by : Miguel Luis
//
// Date       : 15.10.1998
//
//*****************************************************************
// Description:Register declaration for use in C language
//*****************************************************************
#ifndef __XE8000_h
#define __XE8000_h

//-----------------------------------------------------------------
// Register for user or compilator
//-----------------------------------------------------------------
extern volatile char Reg00 __attribute__((page_0));
```

```
extern volatile char Reg01 __attribute__((page_0));
extern volatile char Reg02 __attribute__((page_0));
extern volatile char Reg03 __attribute__((page_0));
extern volatile char Reg04 __attribute__((page_0));
extern volatile char Reg05 __attribute__((page_0));
extern volatile char Reg06 __attribute__((page_0));
extern volatile char Reg07 __attribute__((page_0));

//-----------------------------------------------------------------
// Register system
//-----------------------------------------------------------------
extern volatile char RegSysCtrl    __attribute__((page_0));
extern volatile char RegSysReset   __attribute__((page_0));
extern volatile char RegSysClock   __attribute__((page_0));
extern volatile char RegSysMisc    __attribute__((page_0));
extern volatile char RegSysWd      __attribute__((page_0));
extern volatile char RegSysPre0    __attribute__((page_0));
extern volatile char RegSysPre1    __attribute__((page_0));
extern volatile char RegSysTest1   __attribute__((page_0));
extern volatile char RegSysTest2   __attribute__((page_0));
extern volatile char RegSysTest3   __attribute__((page_0));
extern volatile char RegSysTestAna __attribute__((page_0));
extern volatile char RegSysRcTrim1 __attribute__((page_0));
extern volatile char RegSysRcTrim2 __attribute__((page_0));
extern volatile char RegSysWarm    __attribute__((page_0));

//-----------------------------------------------------------------
// Register Port A
//-----------------------------------------------------------------
extern volatile char RegPAIn       __attribute__((page_0));
extern volatile char RegPADebounce __attribute__((page_0));
extern volatile char RegPAEdge     __attribute__((page_0));
extern volatile char RegPAPullup   __attribute__((page_0));
extern volatile char RegPARes0     __attribute__((page_0));
extern volatile char RegPARes1     __attribute__((page_0));
extern volatile char RegPAEvent    __attribute__((page_0));
extern volatile char RegPATest     __attribute__((page_0));

//-----------------------------------------------------------------
// Register Port B
//-----------------------------------------------------------------
extern volatile char RegPBOut    __attribute__((page_0));
extern volatile char RegPBIn     __attribute__((page_0));
extern volatile char RegPBDir    __attribute__((page_0));
extern volatile char RegPBOpen   __attribute__((page_0));
extern volatile char RegPBPullup __attribute__((page_0));
extern volatile char RegPBAna    __attribute__((page_0));
extern volatile char RegPBTest   __attribute__((page_0));

//-----------------------------------------------------------------
// Register Port C
//-----------------------------------------------------------------
extern volatile char RegPCOut  __attribute__((page_0));
extern volatile char RegPCIn   __attribute__((page_0));
extern volatile char RegPCDir  __attribute__((page_0));
extern volatile char RegPCTest __attribute__((page_0));

//-----------------------------------------------------------------
// Register Port D
//-----------------------------------------------------------------
extern volatile char RegPDOut  __attribute__((page_0));
```

```
extern volatile char RegPDIn   __attribute__((page_0));
extern volatile char RegPDDir  __attribute__((page_0));
extern volatile char RegPDTest __attribute__((page_0));

//-----------------------------------------------------------------
// Register EEPROM interface
//-----------------------------------------------------------------
extern volatile char RegEEP  __attribute__((page_0));
extern volatile char RegEEP1 __attribute__((page_0));
extern volatile char RegEEP2 __attribute__((page_0));
extern volatile char RegEEP3 __attribute__((page_0));

//-----------------------------------------------------------------
// Register Event controler
//-----------------------------------------------------------------
extern volatile char RegEvn         __attribute__((page_0));
extern volatile char RegEvnEn       __attribute__((page_0));
extern volatile char RegEvnPriority __attribute__((page_0));
extern volatile char RegEvnEvn      __attribute__((page_0));

//-----------------------------------------------------------------
// Register Interrupt controler
//-----------------------------------------------------------------
extern volatile char RegIrqHig      __attribute__((page_0));
extern volatile char RegIrqMid      __attribute__((page_0));
extern volatile char RegIrqLow      __attribute__((page_0));
extern volatile char RegIrqEnHig    __attribute__((page_0));
extern volatile char RegIrqEnMid    __attribute__((page_0));
extern volatile char RegIrqEnLow    __attribute__((page_0));
extern volatile char RegIrqPriority __attribute__((page_0));
extern volatile char RegIrqIrq      __attribute__((page_0));

//-----------------------------------------------------------------
// Register UART interface
//-----------------------------------------------------------------
extern volatile char RegUartCtrl  __attribute__((page_0));
extern volatile char RegUartCmd   __attribute__((page_0));
extern volatile char RegUartTx    __attribute__((page_0));
extern volatile char RegUartTxSta __attribute__((page_0));
extern volatile char RegUartRx    __attribute__((page_0));
extern volatile char RegUartRxSta __attribute__((page_0));
extern volatile char RegUartTest  __attribute__((page_0));

//-----------------------------------------------------------------
// Register Counters
//-----------------------------------------------------------------
extern volatile char RegCntA       __attribute__((page_0));
extern volatile char RegCntB       __attribute__((page_0));
extern volatile char RegCntC       __attribute__((page_0));
extern volatile char RegCntD       __attribute__((page_0));
extern volatile char RegCntCtrlCk  __attribute__((page_0));
extern volatile char RegCntConfig1 __attribute__((page_0));
extern volatile char RegCntConfig2 __attribute__((page_0));
extern volatile char RegCntOn      __attribute__((page_0));

//-----------------------------------------------------------------
// Register A/D Convertor
//-----------------------------------------------------------------
extern volatile char RegAcOutLsb __attribute__((page_0));
extern volatile char RegAcOutMsb __attribute__((page_0));
extern volatile char RegAcCfg0   __attribute__((page_0));
```

```
extern volatile char RegAcCfg1   __attribute__((page_0));
extern volatile char RegAcCfg2   __attribute__((page_0));
extern volatile char RegAcCfg3   __attribute__((page_0));
extern volatile char RegAcCfg4   __attribute__((page_0));
extern volatile char RegAcCfg5   __attribute__((page_0));

//---------------------------------------------------------------
// Register D/A Convertor (Signal)
//---------------------------------------------------------------
extern volatile char RegDasInLsb __attribute__((page_0));
extern volatile char RegDasInMsb __attribute__((page_0));
extern volatile char RegDasCfg0  __attribute__((page_0));
extern volatile char RegDasCfg1  __attribute__((page_0));

//---------------------------------------------------------------
// Register D/A Convertor (Bias 1)
//---------------------------------------------------------------
extern volatile char RegDab1In  __attribute__((page_0));
extern volatile char RegDab1Cfg __attribute__((page_0));

//---------------------------------------------------------------
// Register Voltage Multiplier
//---------------------------------------------------------------
extern volatile char RegVmultCfg0 __attribute__((page_0));
extern volatile char RegVmultCfg1 __attribute__((page_0));

//---------------------------------------------------------------
// Register Voltage Low Detect
//---------------------------------------------------------------
extern volatile char RegVld0 __attribute__((page_0));
extern volatile char RegVld1 __attribute__((page_0));

#endif // __XE8000_h
```


Printed in Switzerland

Date of release 07-99

A907-041 - AN8000.01 Application Note: XE8000 address pack

SUNSTAR 商斯达实业集团是集研发、生产、工程、销售、代理经销 、技术咨询、信息服务等为一体的高科技企业，是专业高科技电子产品生产厂家，是具有 10 多年历史的专业电子元器件供应商，是中国最早和最大的仓储式连锁规模经营大型综合电子零部件代理分销商之一,是一家专业代理和分銷世界各大品牌 IC 芯片和電子元器件的连锁经营綜合性国际公司，专业经营进口、国产名厂名牌电子元件，型号、种类齐全。在香港、北京、深圳、上海、西安、成都等全国主要电子市场设有直属分公司和产品展示展销窗口门市部专卖店及代理分销商,已在全国范围内建成强大统一的供货和代理分销网络。 我们专业代理经销、开发生产电子元器件、集成电路、传感器、微波光电元器件、工控机/DOC/DOM 电子盘、专用电路、单片机开发、MCU/DSP/ARM/FPGA 软件硬件、二极管、三极管、模块等，是您可靠的一站式现货配套供应商、方案提供商、部件功能模块开发配套商。商斯达实业公司拥有庞大的资料库，有数位毕业于著名高校——有中国电子工业摇篮之称的西安电子科技大学（西军电）并长期从事国防尖端科技研究的高级工程师为您精挑细选、量身订做各种高科技电子元器件，并解决各种技术问题。

微波光电部专业代理经销高频、微波、光纤、光电元器件、组件、部件、模块、整机；电磁兼容元器件、材料、设备；微波 CAD、EDA 软件、开发测试仿真工具；微波、光纤仪器仪表。欢迎国外高科技微波、光纤厂商将优秀产品介绍到中国、共同开拓市场。长期大量现货专业批发高频、微波、卫星、光纤、电视、CATV 器件： 晶振、VCO、连接器、PIN 开关、变容二极管、开关二极管、低噪晶体管、功率电阻及电容、放大器、功率管、MMIC、混频器、耦合器、功分器、振荡器、合成器、衰减器、滤波器、隔离器、环行器、移相器、调制解调器；光电子元器件和组件：红外发射管、红外接收管、光电开关、光敏管、发光二极管和发光二极管组件、半导体激光二极管和激光器组件、光电探测器和光接收组件、光发射接收模块、光纤激光器和光放大器、光调制器、光开关、DWDM 用光发射和接收器件、用户接入系统光光收发器件与模块、光纤连接器、光纤跳线/尾纤、光衰减器、光纤适 配器、光隔离器、光耦合器、光环行器、光复用器/转换器；无线收发芯片和模组、蓝牙芯片和模组。

更多产品请看本公司产品专用销售网站:

商斯达微波光电产品网:HTTP://www.rfoe.net/

商斯达中国传感器科技信息网：http://www.sensor-ic.com/

商斯达工控安防网：http://www.pc-ps.net/

商斯达电子元器件网：http://www.sunstare.com/

商斯达消费电子产品网://www.icasic.com/

商斯达实业科技产品网://www.sunstars.cn/ *射频微波光电元器件销售热线:*

地址：深圳市福田区福华路福庆街鸿图大厦 1602 室

电话：0755-83396822 83397033 83398585 82884100

传真：0755-83376182 （0）13823648918 MSN: SUNS8888@hotmail.com

邮编：518033 E-mail:szss20@163.com QQ: 195847376

深圳赛格展销部：深圳华强北路赛格电子市场 2583 号 电话：0755-83665529 25059422

技术支持: 0755-83394033 13501568376

欢迎索取免费详细资料、设计指南和光盘 ；产品凡多，未能尽录，欢迎来电查询。

北京分公司：北京海淀区知春路 132 号中发电子大厦 3097 号

TEL：010-81159046 82615020 13501189838 FAX：010-62543996

上海分公司：上海市北京东路 668 号上海赛格电子市场 D125 号

TEL：021-28311762 56703037 13701955389 FAX：021-56703037

西安分公司：西安高新开发区 20 所(中国电子科技集团导航技术研究所)

西安劳动南路 88 号电子商城二楼 D23 号

TEL：029-81022619 13072977981 FAX:029-88789382