

6 THE ETRAX 4

6.1 Introduction

The ETRAX 4 is the processor prior to the ETRAX 100 in the ETRAX family. The differences between the CRIS implementation in the ETRAX 100 and the ETRAX 4 are presented in this chapter.

6.2 Special Registers

The processor architecture defines 16 *Special Registers* (P0 - P15), ten of which are implemented in the ETRAX 4. The special registers in the ETRAX 4 are:

MNEM	Reg.No.	Description	Width
VR	P1	Version register	8 bits
CCR	P5	Condition Code Register	16 bits
DCR0	P6	DMA Channel 0 Count Register	16 bits
DCR1	P7	DMA Channel 1 Count Register	16 bits
IBR	P9	Interrupt Base Register	32 bits
IRP	P10	Interrupt Return Pointer	32 bits
SRP	P11	Subroutine Return Pointer	32 bits
DTP0	P12	DMA Channel 0 Transfer Pointer	32 bits
DTP1	P13	DMA Channel 1 Transfer Pointer	32 bits
BRP	P14	Breakpoint Return Pointer	32 bits

Table 6-1 Special registers

Special registers for the ETRAX 4:

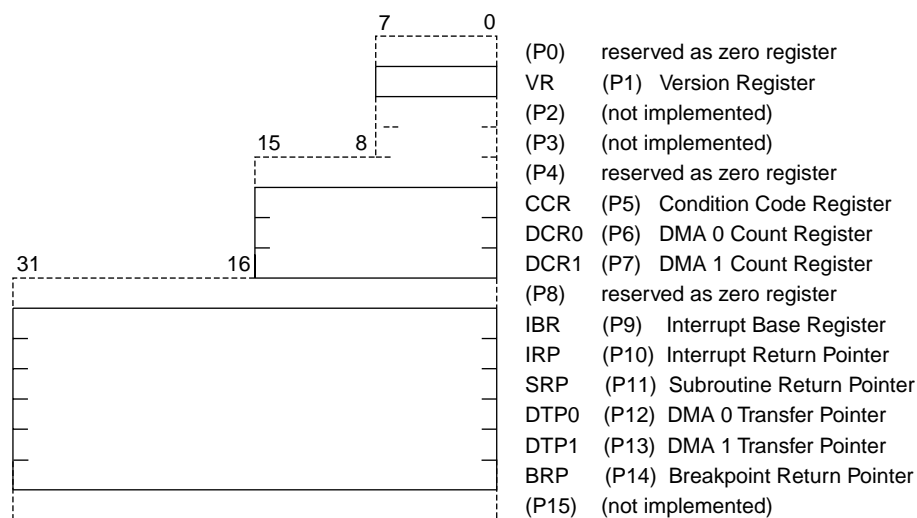


Figure 6-1 Special registers

6.3 Flags and Condition Codes

The ETRAX 4 condition code register (CCR) has no M and B flags, but has instead a D and E flag:

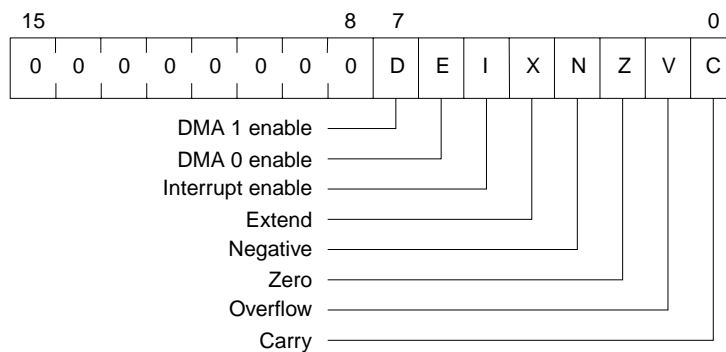


Figure 6-2 The ETRAX 4 Condition Code Register (CCR)

The DCCR register is not available in the ETRAX 4 (this affects the MOVE (to Pd) and POP (to Pd) instructions).

The only difference in the 16 condition codes in table 6-2 below is the EXT (external pin); all other condition codes are the same.

Code	Alt	Condition	Encoding	Boolean function
CC	HS	Carry Clear	0000	\bar{C}
CS	LO	Carry Set	0001	C
NE		Not Equal	0010	\bar{Z}
EQ		Equal	0011	Z
VC		Overflow Clear	0100	\bar{V}
VS		Overflow Set	0101	V
PL		Plus	0110	\bar{N}
MI		Minus	0111	N
LS		Low or Same	1000	C + Z
HI		High	1001	$\bar{C} * \bar{Z}$
GE		Greater or Equal	1010	$N * V + \bar{N} * \bar{V}$
LT		Less Than	1011	$N * \bar{V} + \bar{N} * V$
GT		Greater Than	1100	$N * V * \bar{Z} + \bar{N} * \bar{V} * \bar{Z}$
LE		Less or Equal	1101	$Z + N * \bar{V} + \bar{N} * V$
A		Always True	1110	1
EXT		External Pin	1111	External input

Table 6-2 The ETRAX 4 condition codes

This is the order in which the flags are listed in chapter 1 *Instruction Set Description* (see section 2.5 *Instructions in Alphabetical Order*):

ETRAX 100: M B I X N Z V C

ETRAX 4: D E I X N Z V C

By replacing the M and B flags in the instruction set to D and E, the list of what flags are affected is the same for the ETRAX 4 CPU, except for these instructions:

Instruction		Flags affected	
		D	E
CLEARF		*	*
MOVE to Pd	(Pd != CCR)	-	-
	(Pd == CCR)	-	-
POP		-	-

Table 6-3 Changes in affected flags for the ETRAX 4

6.4 Data Organization in Memory

The ETRAX 4 CPU can operate with an 8-bit or 16-bit wide data bus. Figure Figure 6-3 shows an example of data organization with an 8-bit bus. The same example, but with a 16-bit bus, is shown in figure Figure 6-4.

Example of a structure layout:

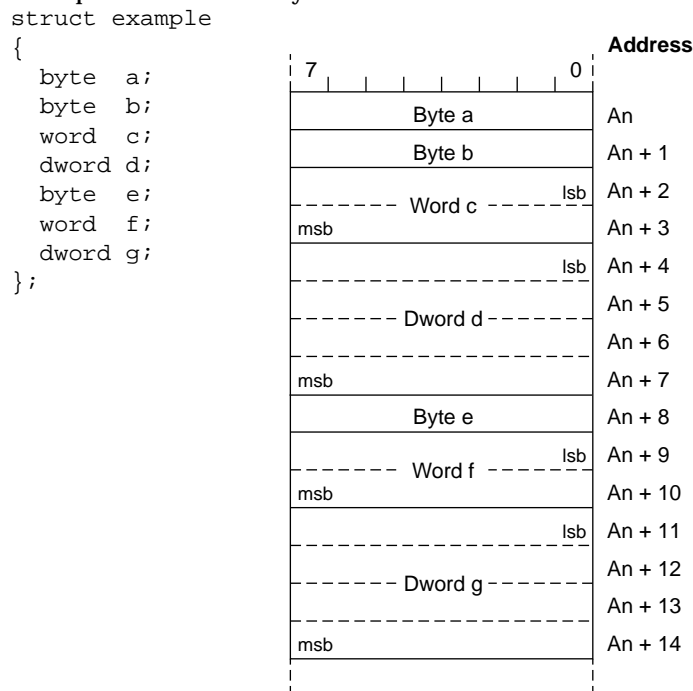


Figure 6-3 Data organization with an 8-bit bus

```

struct example
{
  Byte a;
  Byte b;
  Word c;
  Dword d;
  Byte e;
  Word f;
  Dword g;
};

```

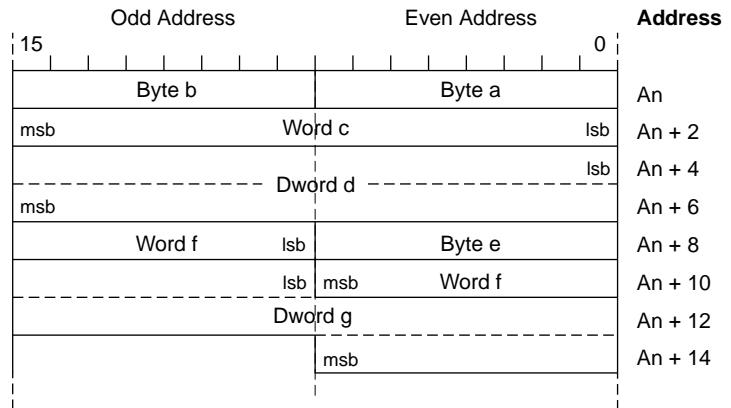


Figure 6-4 Data organization with a 16-bit bus

When a word or a dword is placed at odd addresses in a 16-bit memory (like word f and dword g in figure Figure 6-4), each access to the data will require extra bus cycles. For maximum performance in 16-bit systems, it is recommended to keep word and dword data aligned to even addresses as much as possible.

6.5 Branches, Jumps and Subroutines

The EXT condition is not available in the ETRAX 100 (see table Table 6-2, “The ETRAX 4 condition codes,” on page 164).

6.6 Interrupts and Breakpoints in the ETRAX 4

Only bits 29 and 30 of the Interrupt Base Register (IBR) are implemented in the ETRAX 4, the remaining bits are always zero.

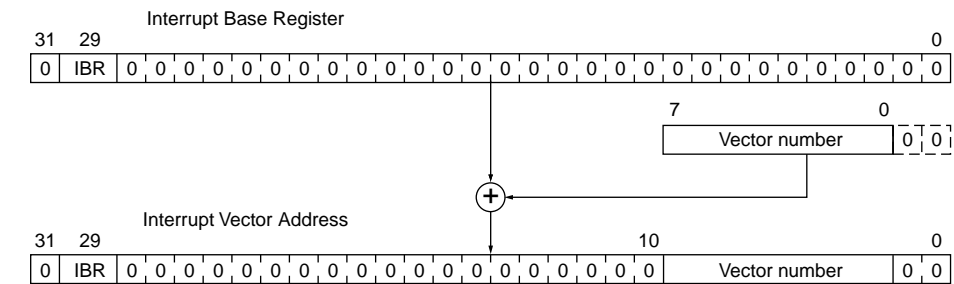


Figure 6-5 Interrupt vector address calculation in the ETRAX 4

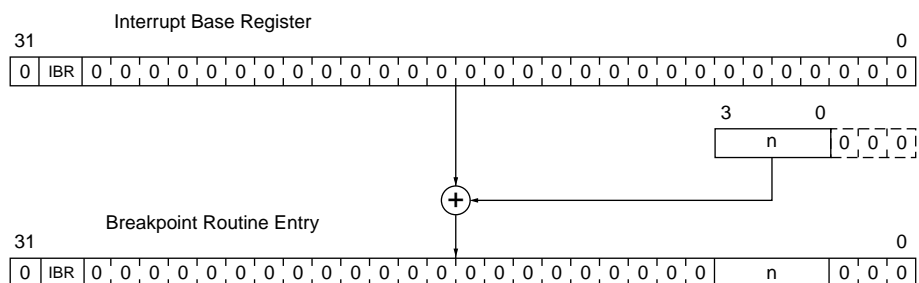


Figure 6-6 Software breakpoint address calculation in the ETRAX 4

Hardware breakpoints are not implemented in the ETRAX 4.

6.7 Reset in the ETRAX 4

6.7.1 Normal case

After reset, the ETRAX 4 CPU starts the execution at address 00000002. The following registers are initialized after reset:

Register	Value (hex)
VR	3
CCR	0000
DCR0	1000
IBR	00000000
DTP0	00000002

Table 6-7 Initialization values of registers after reset in the ETRAX 4

All other registers have unknown values after reset.

6.7.2 Automatic program download

When the automatic program download ("flash-load") is enabled, the initial values of DCR0 and DTP0 change. After the completion of the program download, the registers have the following values:

Register	Value (hex)
VR	3
CCR	0000
DCR0	0000
IBR	00000000
DTP0	40001002

Table 6-8 Initialization values of registers after automatic program download in the ETRAX 4

After the automatic program download, the ETRAX 4 CPU starts to execute at address 40000002 (hex) instead of 00000002.

6.8 DMA

In the ETRAX 100, the DMA is not a part of the CPU but a separate module on the chip. In the ETRAX 4, however, the DMA is an integrated part of the CPU.

6.8.1 The ETRAX 4 DMA

The ETRAX 4 CPU contains two DMA channels. Each channel has a 32-bit *DMA Transfer Pointer* (DTP), a 16-bit *DMA Count Register* (DCR), and a DMA enable flag (*D* or *E*). The connection of each channel to a physical I/O channel is described in the *ETRAX Data Sheet*.

To start a DMA transfer, the DTP of the channel is loaded with the start address of the data block to be transferred, and the DCR of the channel is loaded with the number of transfers. (Loading the DCR with zero will give 65 536 transfers). The DMA enable flag of the channel is then set with the SETF instruction.

For each transfer, the DTP is incremented by one (byte transfer) or two (word transfers), and the DCR is decremented by one. When the DCR counts down to zero, the DMA enable flag is set to zero and the transfers stop.

The DMA can be stopped and started at any time by clearing the DMA enable flag. Note that the SETF and CLEARF instructions are the only instructions that will affect the D and E flags. When CCR is updated using the MOVE instruction, the D and E flags are left unchanged.

DMA channel 0 is designed to be able to automatically load a program ("flash-load") to the system RAM after power up. This feature is enabled by keeping the external FLASH_ input low during reset. In this case, 4096 Bytes (1000 hex) are transferred to the system RAM area with start at address 40000002 hex, before the CPU starts to execute.

6.9 Instruction Set

The ETRAX 4 CPU has a few less instructions than the ETRAX 100 CPU. The instructions which are not available in the ETRAX 4 are:

JBRC, JIRC, JSRC, SWAP

6.9.1 Differences in the instructions

The following instructions are different in the ETRAX 4 compared to the same instructions in the ETRAX 100:

MOVEM (from memroy)	Move to multiple registers from memory	page -67
MOVEM (to memory)	Move from multiple registers from memory	page -68

In autoincrement addressing mode, the address ($si + 4 * \langle \text{number of loaded registers} \rangle$) is loaded to the specified register. For the ETRAX 4 this also applies to the indexed with assign and offset with assign addressing modes. This is different from the ETRAX 100 and from other instructions where the address is stored before the increment.

PUSH (from Ps) Push special register onto stack page -82

The following size information applies to the ETRAX 4:

Size is set according to the size of the pushed register:

11	Byte (Ps = VR)
10	Word (Ps = CCR, DCR0 or DCR1)
00	Dword (Ps = BRP, IBR, IRP, SRP, DTP0 or DTP1)

Table 6-9

CLEARF Clear flags page -40

SETF Set flags page -88

The two most significant bits are D and E:

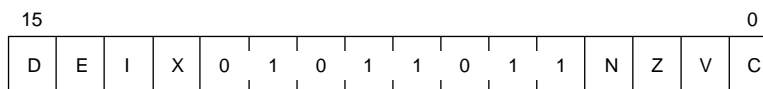


Figure 6-7

6.10 Execution Times for the ETRAX 4

6.10.1 Introduction

Instruction execution times for all CRIS instructions and addressing modes are given below in numbers of CPU cycles. With no wait states, each bus cycle requires two system clock cycles. One system clock cycle is added for each wait state.

6.10.2 Instruction execution times

This section gives the execution times for instructions with the four basic addressing modes: Quick immediate, Register, Indirect, and Autoincrement. Except for the following four special cases, the execution time is the same for all instructions with the same addressing mode and data size.

General case:

Addressing mode	Data size	Data alignment	Execution time	
			16-bit bus	8-bit bus
Quick immediate	6-bit	N/A	1	2
Register	any	N/A	1	2
Indirect, Autoinc.	Byte	any	2	3
Indirect, Autoinc.	Word	even address	2	4
Indirect, Autoinc.	Word	odd address	3	4
Indirect, Autoinc.	Dword	even address	3	6
Indirect, Autoinc.	Dword	odd address	5	6

*Table 6-8***Special case 1:****Bcc instruction**

Branch offset size	Execution time	
	16-bit bus	8-bit bus
Byte	1	2
Word	2	4

*Table 6-9***Special case 2:****MOVEM instruction**

Addressing mode	Data size	Data alignment	Execution time	
			16-bit bus	8-bit bus
Indirect, Autoinc.	Dword	even address	$2n + 1$	$4n + 2$
Indirect, Autoinc.	Dword	odd address	$4n + 1$	$4n + 2$

Table 6-10

(Where n is the number of registers moved.)

Special case 3:**PC operand**

One idle bus cycle is added to the execution times given above, if PC is used as the destination operand in any of the following instructions:

ABS	ADD	ADDQ	ADDS	ADDU	AND
ANDQ	ASR	ASRQ	BTSTQ	MOVEM	MOVEQ
MOVE except from a special register			MOVS	MOVU	OR
ORQ	POP	SUB	SUBQ	SUBS	SUBU
XOR					

One idle bus cycle is also added for the TEST.m PC instruction.

Special case 4:**Break instruction**

The BREAK instruction takes two cycles to execute on a 16-bit data bus, and three cycles on an 8-bit data bus.

6.10.3 Complex addressing modes execution times

The table below gives the extra execution time required to calculate the effective address in complex addressing modes. The effective address calculation time is added to the Indirect/Autoincrement execution time given in 6.10.2 *Instruction execution times* to give the total execution time of the instruction.

Addressing mode	Data alignment	Execution time	
		16-bit bus	8-bit bus
Indexed	N/A	1	2
Indexed with assigned	N/A	1	2
Immediate Byte offset	N/A	1	2
Indirect Byte offset	any	2	3
Word offset	even address	2	4
Word offset	odd address	3	4
Dword offset	even address	3	6
Dword offset	odd address	5	6
Immediate Byte offset with assign	N/A	1	2
Indirect Byte offset with assign	any	2	3
Word offset with assign	even address	2	4
Word offset with assign	odd address	3	4
Dword offset with assign	even address	3	6
Dword offset with assign	odd address	5	6
Double indirect	even address	3	6
Double indirect	odd address	5	6
Double indirect with autoincrement	even address	3	6
Double indirect with autoincrement	odd address	5	6
Absolute	N/A	3	6

Table 6-11

Note: Data alignment refers to the alignment of data involved in the effective address calculation

6.10.4 Interrupt acknowledge execution time

The interrupt acknowledge sequence, including the interrupt acknowledge cycle and the interrupt vector read following it, requires 3 bus cycles on a 16-bit bus, and 5 bus cycles on an 8-bit bus.

6.10.5 DMA transfer execution time

Each DMA transfer requires one bus cycle.