Fautronix GmbH

When complexity seems easy

Xilinx PicoBlaze™ software development with FIDEx™

All Programmable PLC2 Days 2014

Christoph Fauck

www.fautronix.com

Version 0.4

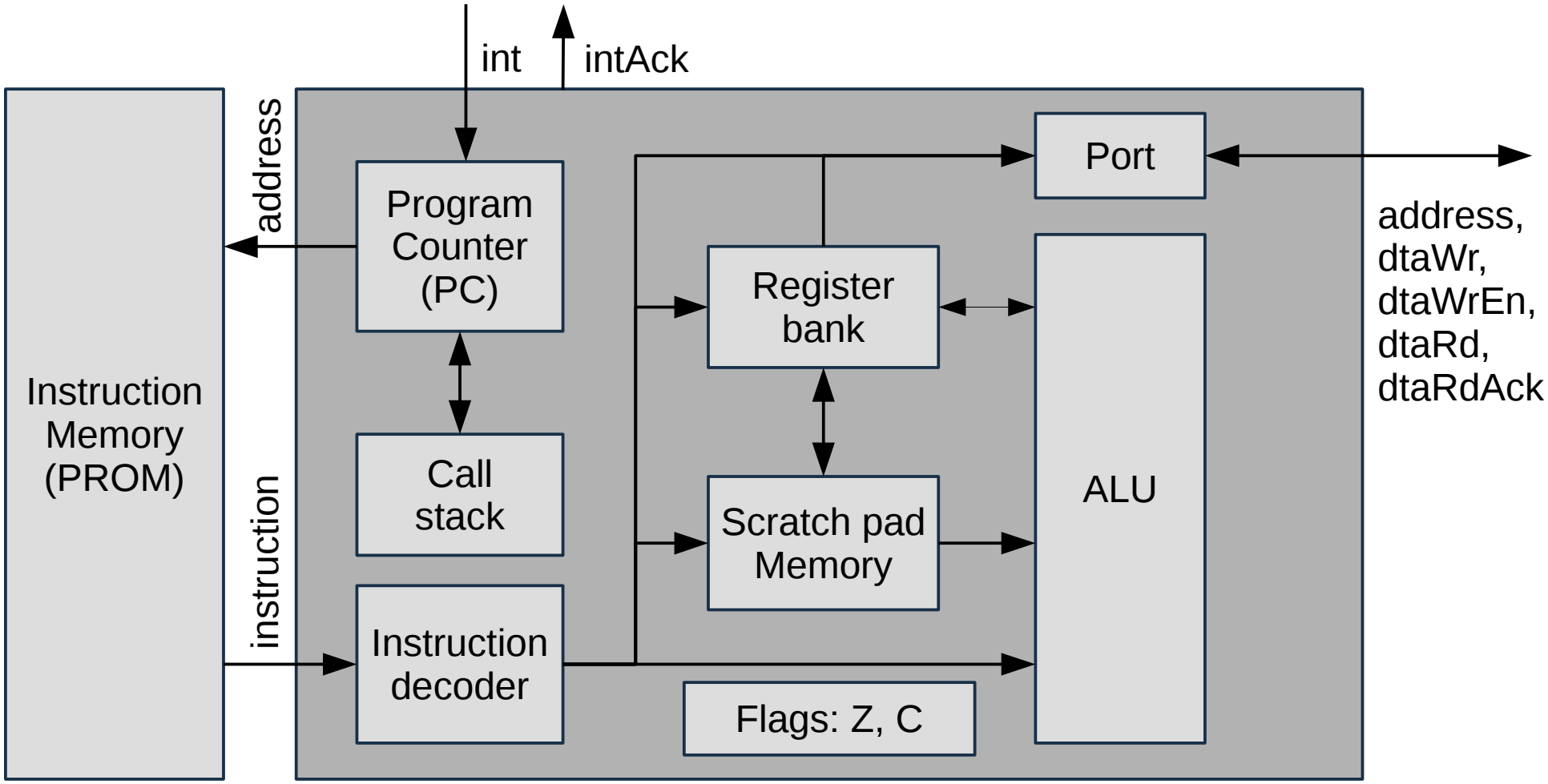
Outline

- Introduction to the PicoBlaze family
- Why using assembler
- Introduction to FIDEx
- Connecting logic to a PicoBlaze
- Using FIDEx with examples

The PicoBlaze family

PicoBlaze family member	Xilinx FPGA / CPLD family
PicoBlaze 6	Spartan-6, Virtex-6 and newer FPGAs
PicoBlaze 3	Spartan-3, Virtex-II, Virtex-II Pro, Virtex 4 and newer FPGAs
PicoBlaze II	Virtex-II FPGAs
PicoBlaze	Virtex-E and Spartan-II/IIE FPGAs
PicoBlaze CPLD	CoolRunner-II CPLDs

The PicoBlaze in detail



PicoBlaze compared

Features	PicoBlaze 6	PicoBlaze 3	PicoBlaze II	PicoBlaze	PicoBlaze CPLD
IO ports	256	256	256	256	256
Register count	2 x 16	16	32	16	8
Scratchpad size in Bytes	64/128/256	64	0	0	0
Callstack size	30	31	31	15	4
Program size in instructions	1024/ 2048/ 4096	1024	1024	256	256
Cycles per instruction	2	2	2	2	2
Interrupt count	1	1	1	1	1

Why using Assembler

- Assembler in VHDL simulator easily traceable, C only troublesome
- C in some things easier and clearer (loops, function calls)
- Limits of the hardware more transparent in assembler (e.g. register count)
- Runtimes clearly defined in assembler, compiler dependent in C
- For very small processors similar coding effort

FIDEx

FIDEx is an IDE for developing manufacturer platform independent assembler code. It contains

- a project manager
- a coding editor
- a source navigator
- a multi-pass assembler supporting several processors
- a simulator supporting several processors
- an extensive online manual
- a message viewer and
- a scripting engine

Project manager

FIDEx

Configuration File Edit Navigation Assembler Simulator Tools Help

Reload Help

Source Navigator

Sort order: Document structure

Bookmarks: Set Clear Clear all

Project Manager | Getting started

Project manager

New | Open

Open project

Project file:

Add to collection | Open

Project collection:

Paging example

Version: 0.1

Comment: A paging example for Xilinx PicoBlaze 3

Project file: ...mplePicoBlaze3/FIDEx/pagingExample.fidex

Remove from collection | Open

Simulator testbench for PicoBlaze 3

Version: 0.1

Close | Cancel

The project manager

The project manager is the core component to generate, open and close projects.

Generate a new project

For generating a new project, please activate the "new" register card of the project manager.

On this register card you have to define the minimum settings for a project:

- a file name with its location
- a project title

After setting the minimum project settings you can generate the new project, add them to the project collection and open it by clicking the corresponding button on the "new" register card.

Open an existing project

For opening an existing project, please activate the "open" register card of the project manager.

After selecting the project file you can open the project or add them to the project collection by clicking the corresponding button on the "open"

EQU list | Help

Content | Index

- Application manual
- Coding manual
- Errata
- Getting started
- Processors

Messages

Message | Help link | Source link

Loading fonts...

Font size: Mode: | Processor: Memory usage: | License: Educational Expire date: 2015-04-07

Main window

FIDEx - pagingExample.fidex (Title: Paging example, Project version: 0.1)

Configuration File Edit Navigation Assembler Simulator Tools Help

Reload Help RST INT DEC HEX OCT

Source Navigator

Sort order: Document structure

Bookmarks: Set Clear Clear all

- Config
  - Editor config
  - PicoBlaze 3 config
  - Instruction memory config
  - IODevices
    - BRAM0
    - BRAM1
    - BRAM2
    - BRAM3
    - BRAM4
    - BRAM5
    - BRAM6
    - BRAM7
    - SCRPD
  - Global hardware mapping
  - Memory config
  - IODevice config
- Shared
  - S::switchToBank:
  - S::b1ExampleFct:
  - S::b2ExampleFct:
  - S::b3ExampleFct:
  - S::b4ExampleFct:
  - S::b5ExampleFct:
  - S::b6ExampleFct:
  - S::b7ExampleFct:
  - S::isr:
- Page 0
  - P0::main:

Getting started | config.psm | hwMap.psm | Shared.psm | page0.psm | page1.psm | page2.psm | page3.p

```
1  ; \navGrpBegin "Shared"
2
3
4  #NAMESPACE S
5
6      #EQU reg_work,      ::reg_work      ; Mapping of local reg_work to global reg_work
7
8      #EQU BANKSEL0,      0x00
9      #EQU BANKSEL1,      0x01
10     #EQU BANKSEL2,      0x02
11     #EQU BANKSEL3,      0x03
12     #EQU BANKSEL4,      0x04
13     #EQU BANKSEL5,      0x05
14     #EQU BANKSEL6,      0x06
15     #EQU BANKSEL7,      0x07
16
17     #EQU addr_bankSel,  0x00
18
19     #ORG MEMRANGE,      SHARED
20
21 ; ***********************************************************************
22 ; Switches between banks
23 ; ***********************************************************************
24 switchToBank:
25
26     ; ***********************************************************************
27     ; Switch between banks
28     ; ***********************************************************************
29     swToB0:         LOAD    reg_work,       BANKSEL0
30                     OUT     reg_work,       addr_bankSel
31                     RET
32
```

EQU list | Help

Content | Index

- Application manual
- Coding manual
- Errata
- Getting started
- Processors

Messages

Message	Help link	Source link
Asselmbling		
Open project: Done		

Font size: 9 pt Mode: Insert Processor: Xilinx PicoBlaze 6 Memory usage: Shar 49 Page 10 Page 3 Page 3 Page 3 Page 3 Page 3 Page 3 Page 3 License: Educational Expire date: 2015-04-07

Online help

FIDEx - pagingExample.fidex (Title: Paging example, Project version: 0.1)

Configuration File Edit Navigation Assembler Simulator Tools Help

Reload Help RST INT DEC HEX OCT

Source Navigator

Sort order: Document structure

Bookmarks: Set Clear Clear all

- Config
  - Editor config
  - PicoBlaze 3 config
  - Instruction memory config
  - IODevices
    - BRAM0
    - BRAM1
    - BRAM2
    - BRAM3
    - BRAM4
    - BRAM5
    - BRAM6
    - BRAM7
    - SCRPD
- Global hardware mapping
- Memory config
- IODevice config
- Shared
  - S::switchToBank:
  - S::b1ExampleFct:
  - S::b2ExampleFct:
  - S::b3ExampleFct:
  - S::b4ExampleFct:
  - S::b5ExampleFct:
  - S::b6ExampleFct:
  - S::b7ExampleFct:
  - S::isr:
- Page 0
  - P0::main:

page1.psm page2.psm page3.psm page4.psm page5.psm page6.psm page7.psm Instruction set

Instruction set

Instruction	Lattice Mico8 v3.x	Xilinx PicoBlaze 6	Xilinx PicoBlaze 3	Xilinx PicoBlaze II	Xilinx PicoBlaze	Xilinx PicoBlaze CPLD
3rd party Mnemonics	Lattice			Xilinx		
Program flow						
CALL label	Yes	Yes	Yes	Yes	Yes	Yes
CALL Z, label	Yes	Yes	Yes	Yes	Yes	Yes
CALL C, label	Yes	Yes	Yes	Yes	Yes	Yes
CALL NZ, label	Yes	Yes	Yes	Yes	Yes	Yes
CALL NC, label	Yes	Yes	Yes	Yes	Yes	Yes
CALL (regX, regY)	No	Yes	No	No	No	No
JUMP label	Yes	Yes	Yes	Yes	Yes	Yes
JUMP Z, label	Yes	Yes	Yes	Yes	Yes	Yes
JUMP C, label	Yes	Yes	Yes	Yes	Yes	Yes
JUMP NZ, label	Yes	Yes	Yes	Yes	Yes	Yes
JUMP NC, label	Yes	Yes	Yes	Yes	Yes	Yes
JUMP (regX, regY)	No	Yes	No	No	No	No
RET	Yes	Yes	Yes	Yes	Yes	Yes
RET Z	No	Yes	Yes	Yes	Yes	Yes
RET C	No	Yes	Yes	Yes	Yes	Yes
RET NZ	No	Yes	Yes	Yes	Yes	Yes
RET NC	No	Yes	Yes	Yes	Yes	Yes
SET C	Yes	No	No	No	No	No

EQU list Help

Content Index

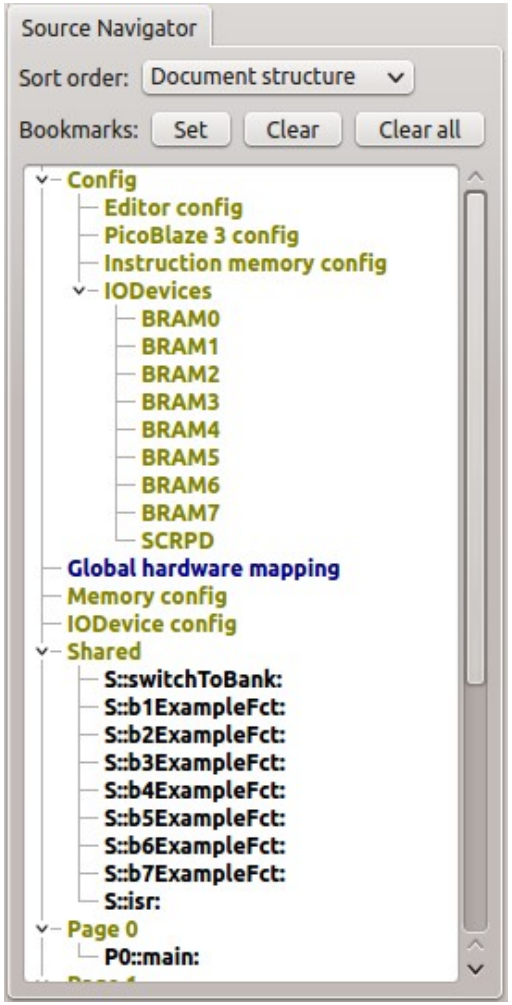
- Application manual
- Coding manual
  - Code Navigator Instructions
  - Directives
  - Formulas
  - Instruction set
  - Numerics
  - Setting up memories
  - Strings
- Errata
- Getting started
- Processors

Messages

Message	Help link	Source link
Asselmbling		
Open project: Done		

Font size: 9 pt Mode: Insert Processor: Xilinx PicoBlaze 6 Memory usage: Share 49 Page 10 Page 3 Page 3 Page 3 Page 3 Page 3 Page 3 Page 3 License: Educational Expire date: 2015-04-07

Source navigator

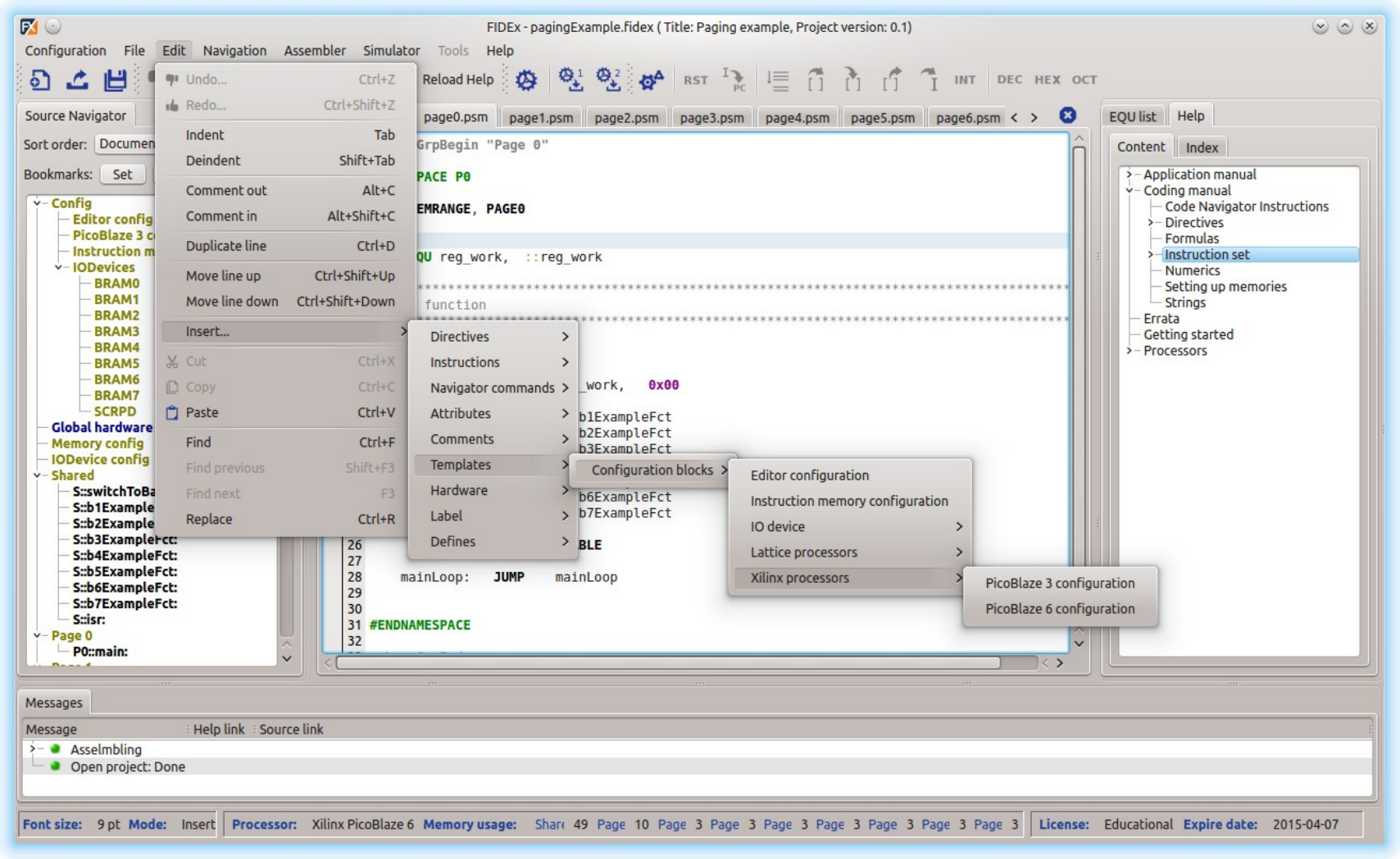


Navigator items:

- Label
- Navigator instructions

```
; \navConst label
; \navConst "Label text"

; \navHw label
; \navHw "Label text"

; \navSep label
; \navSep "Label text"

; \navGrpBegin label
; \navGrpBegin "Label text"

; \navGrpEnd
```

Templates


FIDEx - pagingExample.fidex (Title: Paging example, Project version: 0.1)
Configuration File Edit Navigation Assembler Simulator Tools Help
Undo... Ctrl+Z
Redo... Ctrl+Shift+Z
Indent Tab
Deindent Shift+Tab
Comment out Alt+C
Comment in Alt+Shift+C
Duplicate line Ctrl+D
Move line up Ctrl+Shift+Up
Move line down Ctrl+Shift+Down
Insert...
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Find Ctrl+F
Find previous Shift+F3
Find next F3
Replace Ctrl+R
Directives
Instructions
Navigator commands
Attributes
Comments
Templates
Hardware
Label
Defines
Configuration blocks
Editor configuration
Instruction memory configuration
IO device
Lattice processors
Xilinx processors
PicoBlaze 3 configuration
PicoBlaze 6 configuration
Source Navigator
Bookmarks:
Config
Editor config
IODevices
BRAM0
BRAM1
BRAM2
BRAM3
BRAM4
BRAM5
BRAM6
BRAM7
SCRPD
Global hardware
Memory config
IODevice config
Shared
Page 0
P0::main:
page0.psm page1.psm page2.psm page3.psm page4.psm page5.psm page6.psm
EQU list Help
Content Index
Application manual
Coding manual
Code Navigator Instructions
Directives
Formulas
Instruction set
Numerics
Setting up memories
Strings
Errata
Getting started
Processors
Messages
Message Help link Source link
Asselmbling
Open project: Done
Font size: 9 pt Mode: Insert Processor: Xilinx PicoBlaze 6 Memory usage: License: Educational Expire date: 2015-04-07

Import filter

FIDEx - pagingExample.fidex (Title: All Programmable PLC2 Days 2014 paging example, Project version: 0.1)

Configuration File Edit Navigation Assembler Simulator Tools Help

Source Navigator

Sort order: Document structure

Bookmarks: Set Clear Clear all

- Config
  - Editor config
  - PicoBlaze 3 config
  - Instruction memory config
  - IODevices
    - BRAM0
    - BRAM1
    - BRAM2
    - BRAM3
    - BRAM4
    - BRAM5
    - BRAM6
    - BRAM7
    - SCRPD
- Global hardware mapping
- Memory config
- IODevice config
- Shared
  - S::switchToBank:
  - S::b1ExampleFct:
  - S::b2ExampleFct:
  - S::b3ExampleFct:
  - S::b4ExampleFct:
  - S::b5ExampleFct:

page6.psm page7.psm HEX output files Import files

Import files

Source file

Source file: gs/PLC2-Days-2014/pagingExample/FIDEx/page0.psm

Tab space: 4 Syntax: Xilinx

Imports Xilinx PicoBlaze assen

```
13
14      LOAD reg_wo
15      OUT  reg_
16
```

- Please select an import filter
- Lattice Mico8
- Xilinx
- Mediatronix - pBlazIde
- Fautronix GmbH - openPICIDE until 0.3.x
- Fautronix GmbH - openPICIDE until 0.5.x

Restart Convert line Convert all

Destination file

Tab space: 4

```
13
14      LOAD reg_work, 0x00
```

Use converted file

Import files

FIDEx supports the import of different assembler instructions and coding styles and provides an import filter dialog.

The code to import is parsed and converted to the Idevix assembler by applying filter rules line by line. The import filter are located in a separate directore an may be user adaptable.

Converting a file

The import may be done by doing the following steps:

1. Selecting a source file to import
2. Selecting an import filter
3. Adjusting tab spaces for accurate viewing of the source and destination code
4. Convert the source code into destination code
   - manually line by line
   - completely at one time

EQU list Help

Content Index

- Application manual
- Coding manual
- Errata
- Getting started
- Processors
- The Fautronix GmbH

Messages

Message	Help link	Source link
Asselmbling		
Open project: Done		

Font size: 9 pt Mode: Insert Processor: Xilinx PicoBlaze 3 Memory usage: Share 49 Page 11 Page 3 Page 3 Page 3 Page 3 Page 3 Page 3 Page 3 License: Educational Expire date: 2015-04-07

The FIDEx instruction set

Category	Instruction set
Program flow	call, jump, ret, nop
Logic	and, comp, compC, or, test, testC, xor
Arithmetic	add, addC, sub, subC
Shift and rotate	sl0, sl1, slx, sr0, sr1, srx, rol, rolc, ror, rorc
Move data	load, loadRet, wrMem, rdMem, wrPrt, rdPrt, hwBuild, regBank, star
Interrupt	int enable/disable, reti enable/disable

Directives

FIDEx supports C-style directives for defining constants, renaming processor hardware, manipulating the address mapping and so on:

- EQU
- IF, ELSEIF, ELSE and DEF
- MACRO, ENDMACRO
- NAMESPACE, ENDNAMESPACE
- ORG
- SET

EQU directives

EQU directives are used to

- define and rename constants
- rename processor hardware

```
; Recusive equations
#EQU equRegSubst1,  equRegSubst2
#EQU equRegSubst2,  equSubstNum
#EQU equSubstNum,   0x01

; Renaming hardware components
#EQU yourRegName,     s0
#EQU yourRegBankName, A
```

Project settings dialog

FIDEx - pagingExample.fidex (Title: Paging example, Project version: 0.1)

Configuration File Edit Navigation Assembler Simulator Tools Help

Source Navigator

Sort order: Document structure

Bookmarks: Set Clear Clear all

- Config
  - Editor config
  - PicoBlaze 3 config
  - Instruction memory config
  - IODevices
    - BRAM0
    - BRAM1
    - BRAM2
    - BRAM3
    - BRAM4
    - BRAM5
    - BRAM6
    - BRAM7
    - SCRPD
- Global hardware mapping
- Memory config
- IODevice config
- Shared
  - S::switchToBank:
  - S::b1ExampleFct:
  - S::b2ExampleFct:
  - S::b3ExampleFct:
  - S::b4ExampleFct:
  - S::b5ExampleFct:
  - S::b6ExampleFct:
  - S::b7ExampleFct:
  - S::isr:
- Page 0
  - P0::main:

page4.psm page5.psm page6.psm page7.psm Instruction set Configuration

Info Sources Assembler Processor Editor Code Navi

Assembler configuration

Character set

Default character set: UTF-8

Assembler behavior

Default charset behavior: Warn by using default charset

Equations overwrite behavior: Warn by overwriting equations

Settings overwrite behavior: Overwrite settings without warning

Check interrupt vector behavior: Error if not defined by assembler code

Output file configuration

Output file write events

Events:	Button 1	Button 2	After saving
Write VHDL files:	✓		
Write verilog files:			
Write hex files:			
Write mem files:			

VHDL output file

Set page default values Restore values Close Cancel

Assembler config page

The assembler configuration page provides facilities to control the assembling and the mechanisms writing instruction code and preset memory contents to the targets.

The coverage of the configuration possibilities of this page is a minimal configuration to getting started! The usage of SET directives provides more access to all available configuration parameters. For more Information please refer to page SET directives.

Character set

Selects the default character set used to convert strings from standard utf-16 format into a byte based format. The charsets to be select depends on the used operating system. For more Information please refer to page SET directives.

Assembler behavior

The assembler behavior controls the usage and overwriting of the following directives.

Default charset behavior

Configures the assembler behavior for using a default charset. For more Information about charsets please refer to page SET directives.

- Don't use default charset

Disables the usage of default charsets by using an

Messages

Message	Help link	Source link
Asselmbling		
Open project: Done		

Font size: 9 pt Mode: Insert Processor: Xilinx PicoBlaze 6 Memory usage: Share 49 Page 10 Page 3 Page 3 Page 3 Page 3 Page 3 Page 3 Page 3 License: Educational Expire date: 2015-04-07

SET directives

SET directives are used to

- overwrite configuration dialog parameter
- set additional configuration parameter not accessible from configuration dialog

```
#SET editor::tabWidth, 4
```

SET directive namespaces

All configuration aprameter are grouped into namespaces:

Namespace	Description
asm	Configuration parameter for assembler engine itself
editor	Configuration parameter for the editor
instMem	Configuration parameter for the instruction memory
ioDev	Configuration parameter for IO devices
proc::xpblze6	Configuration parameter for the PicoBlaze 6
proc::xpblze3	Configuration parameter for the PicoBlaze 3 processor
proc::xpblzeII	Configuration parameter for the PicoBlaze II processor
proc::xpblze	Configuration parameter for the PicoBlaze processor
proc::xpblzeCpld	Configuration parameter for the PicoBlaze CPLD processor

ORG directives

ORG directives are used to affect the mapping of the assembled code to the instruction memory.

```
; Set address of next instruction to 1023
#ORG ADDR, 1023

; Set memory range
#ORG MEMRANGE, [SHARED, PAGEn]
```

Namespace directives

FIDEx supports the usage of nested namespaces to encapsulate code segments.

```
#EQU REG_workGlobal, s0

#NAMESPACE A

#EQU REG_workA, ::REG_workGlobal

#NAMESPACE B

#EQU REG_workB, A::REG_workA

LOAD REG_workB,        0x12
LOAD A::REG_workA,     0x34
LOAD ::REG_workGlobal, 0x56

#ENDNAMESPACE
#ENDNAMESPACE

LOAD REG_workGlobal, A::B::REG_workB
```

Ifdef, elseif, else, endif and define directives

For the selected processor an implicite define will be defined:

- PROC::XPBLZE6
- PROC::XPBLZE3
- PROC::XPBLZEII
- PROC::XPBLZE
- PROC::XPBLZECPLD

```
#DEFINE xyzEn

#IFDEF xyzEn

      your if instructions

#ELSEIF PROC::XPblze6

      your elseif instructions

#ELSE

      your else instructions

#ENDIF
```

Numerics

- Hexadecimal: 0x44 or 0X44
- Decimal: 68
- Oktal: \104
- Binary: b1000100 or B1000100

```
; Defining a constant
#EQU const,   0x1234

; Accessing the higher byte via index
LOAD regWork, const[15..8]

; Accessing the higher byte via formula
LOAD regWork, ( const >> 8 ) & 0xFF
```

Endianess

- All numerics have little endianess as default
- Set to big endianess using an attribute:
  - 0x1234'2BE = 0x3412
  - 0x12345678'4BE = 0x78563412
  - 0x1234'4BE = 0x34120000

Formulas

FIDEx supports the usage of formulas. Formulas can be used in directives as well as in instructions.

Several expressions of a formula may be cascaded using brackets.

```
#EQU num,  0x33

#EQU expr, ( ( ( num + 4) * 5 ) -12 ) / 10 ; expr = 0x1A
```

Formulas

Operators with its priorities:

Mnemonic	Priority	Associativity	Description
#EQU const, ~0xFF	highest	right-to-left	Bitwise inversion
#EQU const, 3 * 5		left-to-right	Multiplication
#EQU const, 10 / 2			Division
#EQU const, 10 % 3			Modulo
#EQU const, 2 + 3			Addition
#EQU const, 5 - 3			Subtraction
#EQU const, 0x0F <<			Shift left
#EQU const, 0xF0 >>			Shift right
#EQU const, 0x0F & 0xF0			Bitwise and
#EQU const, 0x55 ^ 0xAA			Bitwise xor
#EQU const, 0x0F | 0xF0	lowest	left-to-right	Bitwise or

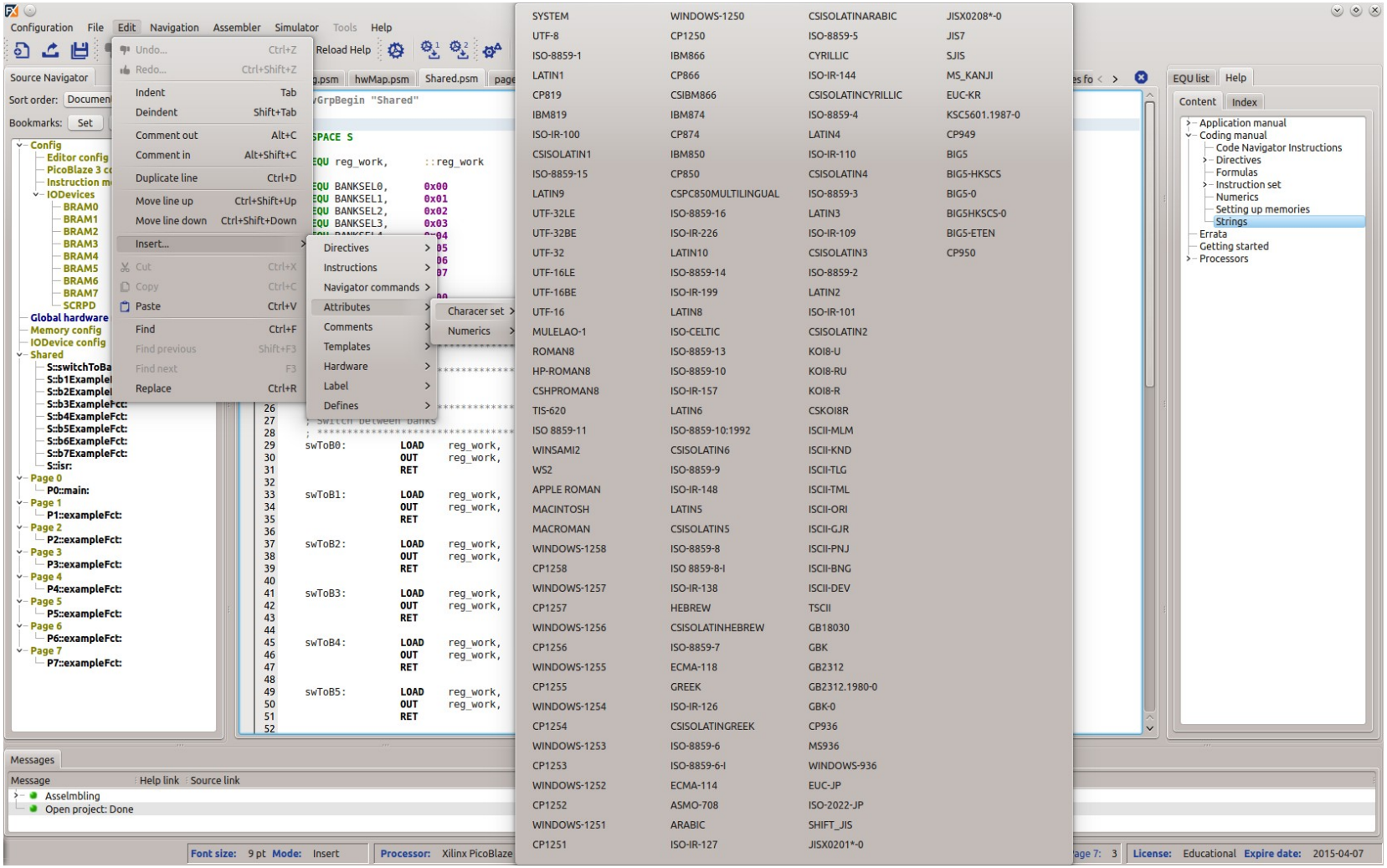
Strings

The use of Strings is subjected to the following rules:

- Enclosing with "-Characters
- Encapsulating "-Characters within the string: \"
- Using a charset/default charset
- Accessing a string using indexes

Character sets are desktop sytem dependent!

Strings - Character sets

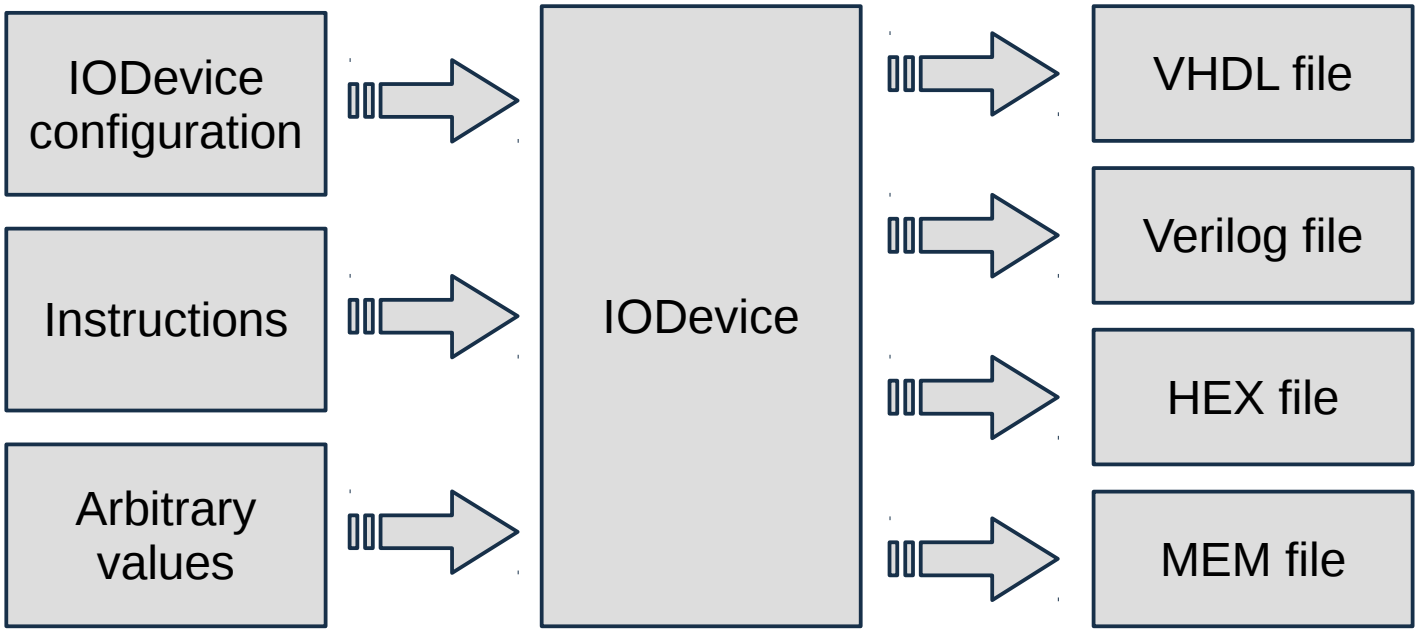


Strings - Examples

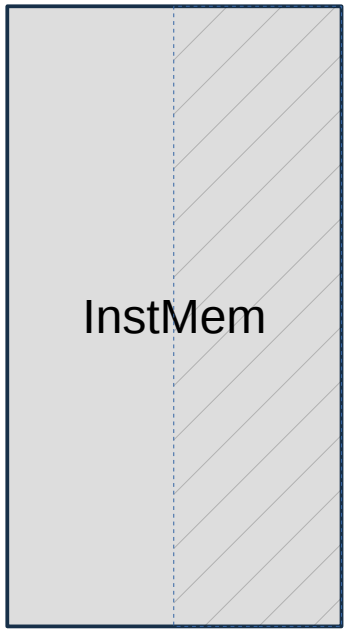
```
#SET ASM::dfltCharset, utf-8

LOAD s0, "A"
LOAD s0, "ö" ; results in an error
```

```
#SET ASM::dfltCharset, utf-8

#EQU yourString "This is a string"

LOAD s0, yourString'len ; s0 = 16
LOAD s1, yourString[0]  ; s1 = 'T'
LOAD s2, yourString[8]  ; s2 = 'a'

#EQU anotherString yourString'ISO-8859-1
```

IO device concept



IO device concept




```
#SET IODev::BRAM0::en,            TRUE
#SET IODev::BRAM0::type,          mem
#SET IODev::BRAM0::size,          1024

#SET IODev::BRAM0::initvalue,     0x00000
#SET IODev::BRAM0::value,         instMem[0:1023][7:0]

#SET IODev::BRAM0::vhdlEn,                TRUE
#SET IODev::BRAM0::vhdlEntityName,        "BRAM0"
#SET IODev::BRAM0::vhdlTmplFile,          "bramTmpl.vhd"
#SET IODev::BRAM0::vhdlTargetFile,        "bram0.vhd"
```

IODevice BRAM0 → VHDL file: bram0.vhd

IODevice BRAM1 → VHDL file: bram1.vhd

IO device base configuration

```
#SET IODev::BRAM0::en,              true
#SET IODev::BRAM0::type,            mem
#SET IODev::BRAM0::size,            1024

#SET IODev::BRAM0::initvalue,       0x00000

#SET IODev::BRAM0::value,           0x01: 0x33
#SET IODev::BRAM0::value,           0x01: 0x44, 55, \66
#SET IODev::BRAM0::value,           instMem
#SET IODev::BRAM0::value,           instMem[page0]
#SET IODev::BRAM0::value,           instMem[0:2047]
#SET IODev::BRAM0::value,           instMem[page1][7:0]
#SET IODev::BRAM0::value,           instMem[0:1023][7:0]
```

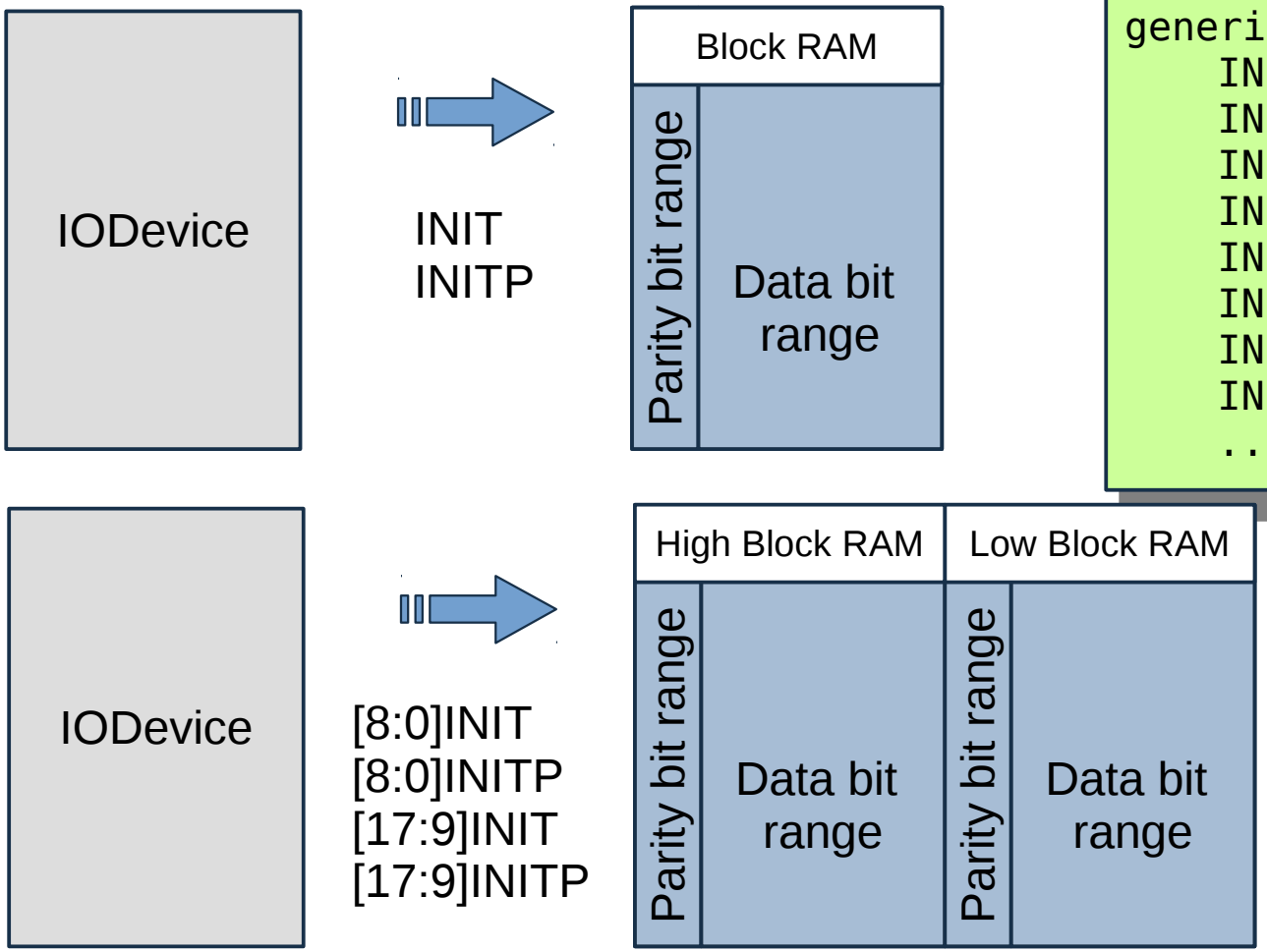
IO device output file configuration

```
#SET IODev::BRAM0::vhdlEn,              true
#SET IODev::BRAM0::vhdlEntityName,      "BRAM0"
#SET IODev::BRAM0::vhdlTmplFile,        "ROM_1bank_form.vhd"
#SET IODev::BRAM0::vhdlTargetFile,      "../VHDL/BRAM0.vhd"

#SET IODev::BRAM0::verilogEn,           true
#SET IODev::BRAM0::verilogEntityName,   "BRAM0"
#SET IODev::BRAM0::verilogTmplFile,     "ROM_1bank_form.v"
#SET IODev::BRAM0::verilogTargetFile,   "BRAM0.v"

#SET IODev::BRAM0::hexEn,               true
#SET IODev::BRAM0::hexTargetFile,       "BRAM0.hex"

#SET IODev::BRAM0::memEn,               true
#SET IODev::BRAM0::memTargetFile,       "BRAM0.mem"
#SET IODev::BRAM0::memGapStuffing,      true
```

Mapping an IO device to VHDL or verilog files

- Xilinx VHDL or Verilog template file
- Parity bit used as Data bit
- Supported Placeholder:
  - {begin template}
  - {INIT_nn}, {INITP_nn}, ...
  - {projectTitle}
  - {projectComment}
  - {projectVersion}
  - {projectPath}
  - {projectFileName}
  - {entityName}
  - {timestamp}

Mapping an IO device to VHDL or verilog files




```
bram: RAMB16_S18
generic map (
    INIT_00 => X"{INIT_00}",
    INIT_01 => X"{INIT_01}",
    INIT_02 => X"{INIT_02}",
    INIT_03 => X"{INIT_03}",
    INIT_04 => X"{INIT_04}",
    INIT_05 => X"{INIT_05}",
    INIT_06 => X"{INIT_06}",
    INIT_07 => X"{INIT_07}",
    ...
```

Simulating

FIDEx - simTestXPBlze6.fidex (Title: Simulator testbench for PicoBlaze 6, Project version: 0.1)

Configuration File Edit Navigation Assembler Simulator Tools Help

Source Navigator | Processor core

PC: 014 PAGE0 HWBuild: 00

Carry 0 Zero 1 Int

Bank: A

s0	33	REG_intReg
s1	00	
s2	FF	REG_work
s3	00	
s4	00	
s5	00	
s6	00	
s7	00	
s8	00	
s9	00	
sA	00	
sB	00	
sC	00	
sD	00	
sE	00	REG_sX
sF	00	REG_sY

0x00	00	00	00	00	00	00	00	00
0x08	00	00	00	00	00	00	00	00
0x10	00	00	00	00	00	00	00	00
0x18	00	00	00	00	00	00	00	00
0x20	00	00	00	00	00	00	00	00
0x28	00	00	00	00	00	00	00	00
0x30	00	00	00	00	00	00	00	00
0x38	00	00	00	00	00	00	00	00

Getting started | simTestXPBlze3.psm | config.psm

```
152                     LOAD    REG_sX,         0xFF
153                     AND     REG_sX,         0x00
154
155                     LOAD    REG_sX,         0xFF
156                     AND     REG_sX,         0xFF
157
158     ; ***********************************************************************
159     ; * Verify AND sX,sY
160     ; ***********************************************************************
161                     LOAD        REG_sX,         0xFF
162                     LOAD        REG_sY,         0x00
163                     AND         REG_sX,         REG_sY
164
165                     LOAD        REG_sX,         0xFF
166                     LOAD        REG_sY,         0xFF
167                     AND         REG_sX,         REG_sY
168
169     ; ***********************************************************************
170     ; * Verify OR sX,kk
171     ; ***********************************************************************
172                     LOAD        REG_sX,         0xFF
173                     OR          REG_sX,         0x00
174
175                     LOAD        REG_sX,         0x00
176                     OR          REG_sX,         0xFF
177
178                     LOAD        REG_sX,         0x00
179                     OR          REG_sX,         0x00
180
181     ; ***********************************************************************
182     ; * Verify OR sX,sY
```

EQU list | Help | Processor ports

Fifo Add Remove all

Addr: 45 In 00

1	33
2	44
3	66
4	77
5	02
6	02
7	02
8	02
9	02
10	02

Addr: 00 IO FF 1 1 1 1 1 1 1 1

Addr: 00 BUSADDR_dbg IO 33 wrTo FF 16

0x0FD	01
0x0FE	01
0x0FF	33
0x100	01
0x101	01
0x102	01
0x103	01
0x104	01

Messages | Processor callstack

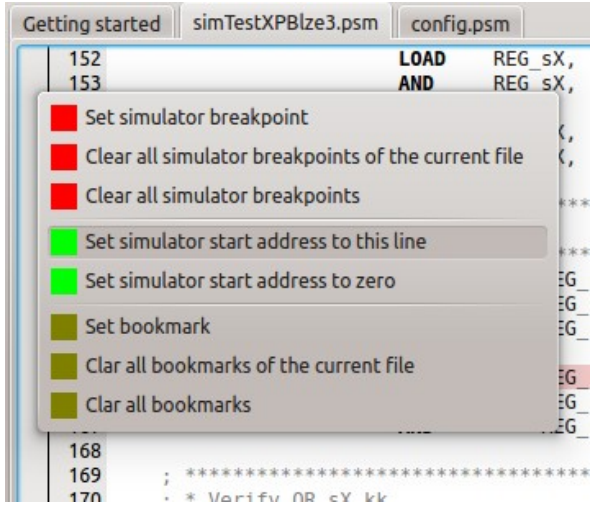
Level	Program counter	Carry flag	Zero flag	Called label
0	C	0	0	ISR
1	16C	1	0	verifyLogic

Font size: 9 pt Mode: Insert | Value format: HEX Clock count: 58 clocks Time count: 725 ns Clear | License: Educational Expire date: 2015-04-07

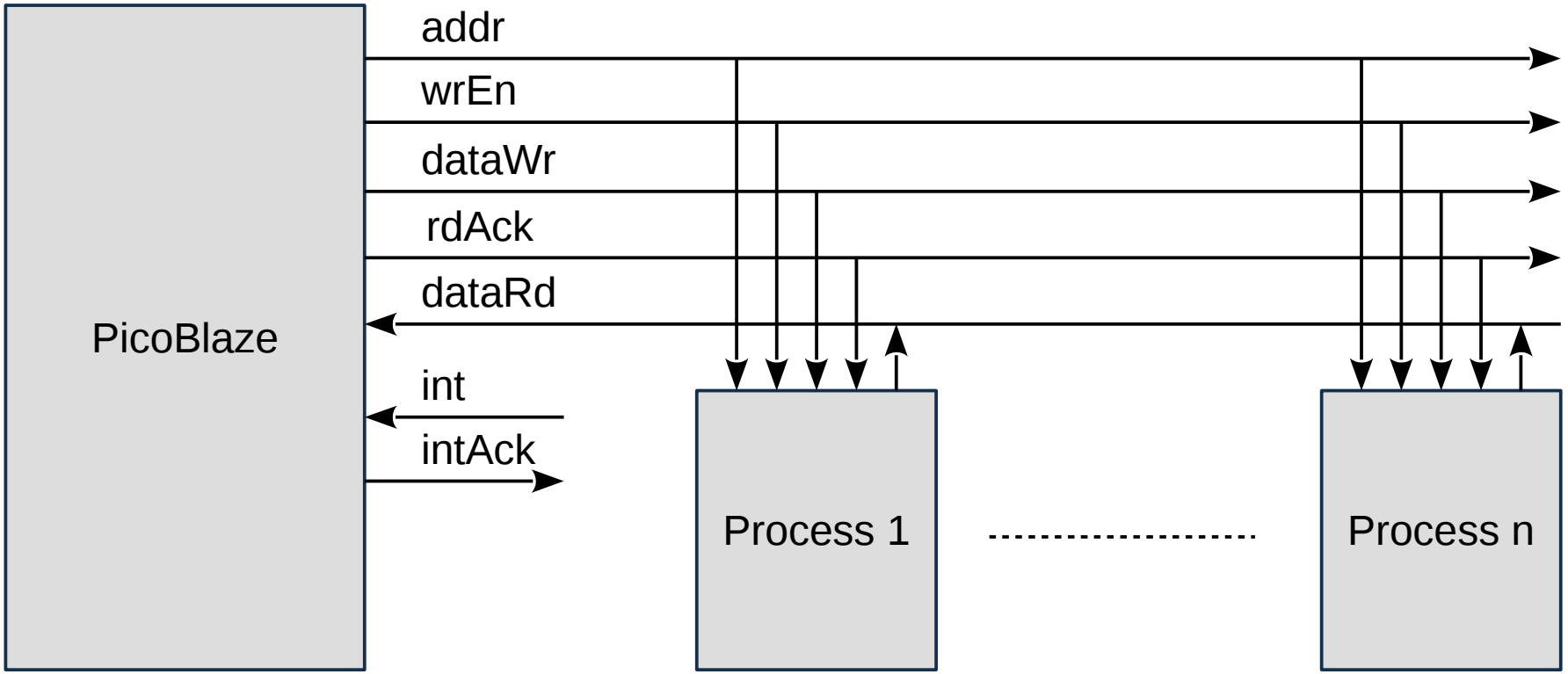
Simulating

Simulation control

- Reset
- Set program counter from text cursor
- Simulate continuous (Run)
- Simulate next line
- Simulate next instruction
- Simulate to end of call (until next RET)
- Simulate to cursor position
- Interrupt



Connecting logic to a PicoBlaze



PicoBlaze to Logic

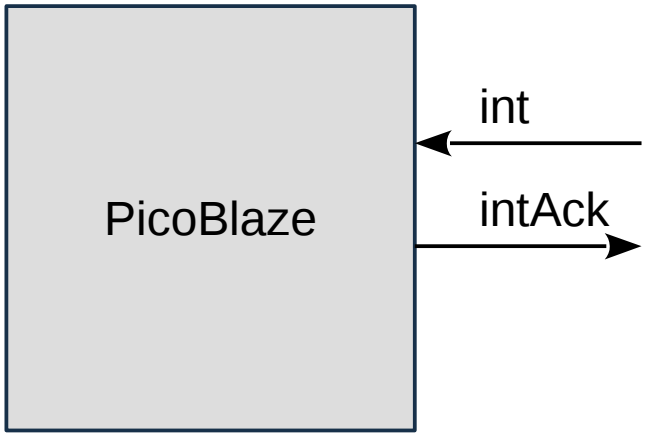
```
selPage: process( in_clk )
begin
    if rising_edge( in_clk )
    then
        if in_rst = '1'
        then
            sig_instrPageSel    <= ( others => '0' );
        else
            if sig_busAddr = X"00" and sig_busWrEn = '1'
            then
                sig_instrPageSel <= sig_busWrDta(3 downto 0);
            end if;
        end if;
    end if;
end process selPage;
```

Logic to PicoBlaze

```
statusReg: process( in_clk )
    variable var_status     : std_logic_vector(7 downto 0);
begin
    if rising_edge( in_clk ) then
        if in_rst = '1' then
            var_status  := ( others => '0' );
        else
            sig_busRdDta    <= ( others => ['0' | 'Z'] );

            if sig_busAddr = X"01" then

                sig_busRdDta    <= var_status;

                if sig_busRdAck = '1' then
                    var_status  := ( others => '0' );
                end if;
            end if;

            var_status  := var_status or in_status;
        end if;
    end if;
end process statusReg;
```

Interrupts

- You can choose whether you want to use intAck.
- The interrupt does not need to be cleared using intAck. It may applied during the complete interrupt service routine.



Interrupts

```
intReg: process( in_clk )
begin
    if rising_edge( in_clk )
    then
        if in_rst = '1'
        then
            sig_int     <= '0';
        else
            sig_int         <= '0';
            sig_busRdDta    <= ( others => '0' );

            if in_int /= X"00" then
                sig_int <= '1';
            end if;

            if sig_busAddr = X"01" then
                sig_busRdDta    <= in_int;
            end if;
        end if;
    end if;
end process intReg;
```

Miscellaneous

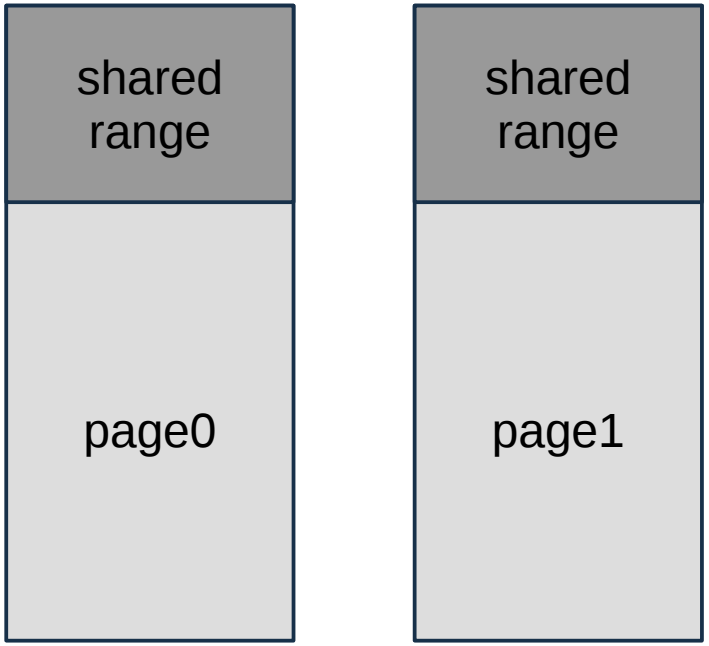
- Saving power
  - „sleep“ signal since PicoBlaze 6
  - lowPowerLoop: JUMP lowPowerLoop
- Address bus sensitive to high fanout → reducing address fanout using auto increment mechanisms
- Organizing bus addresses using a package
- Block oriented assembler coding and documentation for better overview

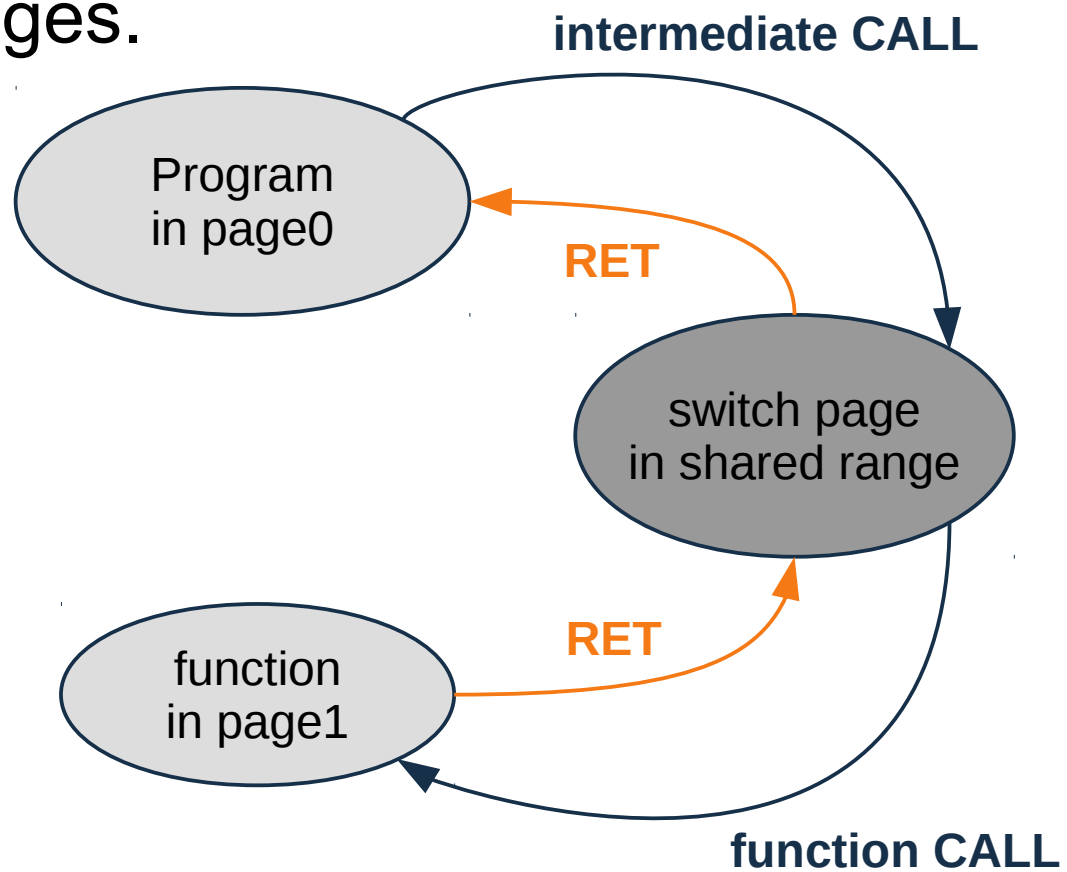
Generating a new project

- Generate a new project
- Generate a new source file
- Inserting a PicoBlaze skeleton template
- Add file to project during first saving
- Open config dialog and select a PicoBlaze processor

Using pages

One of the extended capabilities of Fidex is the support of instruction memory pages.





Page application examples

- Memory expansion if your program exceeds the PicoBlaze supported default memory limit
- Loading context dependent program parts from external memory to pages

Getting Help

- Online Help
- FIDEx forum on www.fautronix.com
- Otherwise contact us directly, please

Example files

All example files used in this presentation can be downloaded from our website

http://www.fautronix.com/fidex-downloads

Change log

Version 0.4

- Added this change log
- Fixed Renaming hardware components bug on page 16
- Fixed Endianess bug on page 24