

,

_____ ()

_____ ()

_____ ()

_____ ()

:

II

,

-18-3

_____ (,)

123 -

,

_____ ()

-

_____ (- -)

_____ ()

:

_____ (; ,)

_____ () _____ (,)

,

()

123 – ’

_____ ()

_____ - -)

_____ ()

:

_____ ()
“ ” _____ 20__ .

_____ (, ,)

1.

“ 30 ” _____ 2020 . _____ 478

2.

_____ 18 _____ 2020 .

3.

VHDL

4.

_____ , _____

5. _____ , _____ , _____ , _____ ,
 () 14 _____

6. _____ (_____ , _____ , _____ , _____ , _____)
 .1) _____

	(_____ , _____ , _____ , _____ , _____)		

1		31.03.20-12.04.20	
	,		
2		13.04.20-20.04.20	
3		21.04.20-27.04.20	
4		28.04.20-05.05.20	
		06.05.20-11.05.20	
6		12.05.20-13.05.20	

30 2020 .

 ()
 | _____ () _____ (; ; ;)
 _____ () _____ (; ; ;)

: 75 ., 19 ., 2 ., 20

.

, VHDL,

,

,

.

,

.

.

;

.

,

.

ABSTRACT

Master's thesis: 75 pages, 19 figures, 2 appendices, 20 sources.

SOC, COMPUTING SYSTEM, SPECIFICATION, VHDL,
ARCHITECTURE, METHOD.

The purpose of the certification work is to study the methods of creating architectural specifications of systems on the crystal and their verification.

As a solution to the tasks, a new method for the development of architectural specifications of systems on the chip and their use to create high-level models of the system. To simplify the creation of executable SOC models, the set of methods includes a method of automating the translation of specifications into executable models in formal languages; and for early joint analysis of various components of the system and protocols of its operation, the developed method involves the use of various methods of data verification at the stage of creating a specification. In addition, the developed set of methods also allows to improve the quality of the created specifications without significant additional efforts of architects and to automate the re-use of specifications of computer systems.

	,	,	,	
			8
			9
1				12
1.1			12
1.2			14
1.3			16
1.4			20
2				21
2.1				21
2.2		-		23
2.3	VHDL		-	29
3				34
3.1				35
3.2			37
3.3			38
3.4			39
3.5			42
3.6			45
4				
				47
4.1			47
4.2			52
4.3				52
4.4			59
			63
			64

.....	66
.....	66

, , ,

RTL –

UML –

–

–

–

, -
 ,
 . - () .
 - () , , ,
 . , ,
 : , ,
 , / , ,
 . : ,
 , - .
 ,
 .
 , .
 , .
 ,
 .
 , , .
 , , .
 , .

:

.

, - ,

, - ,

,

. ,

,

,

.

. - ,

,

,

,

. - ,

,

,

.

,

.

-

.

.

:

-

;

-

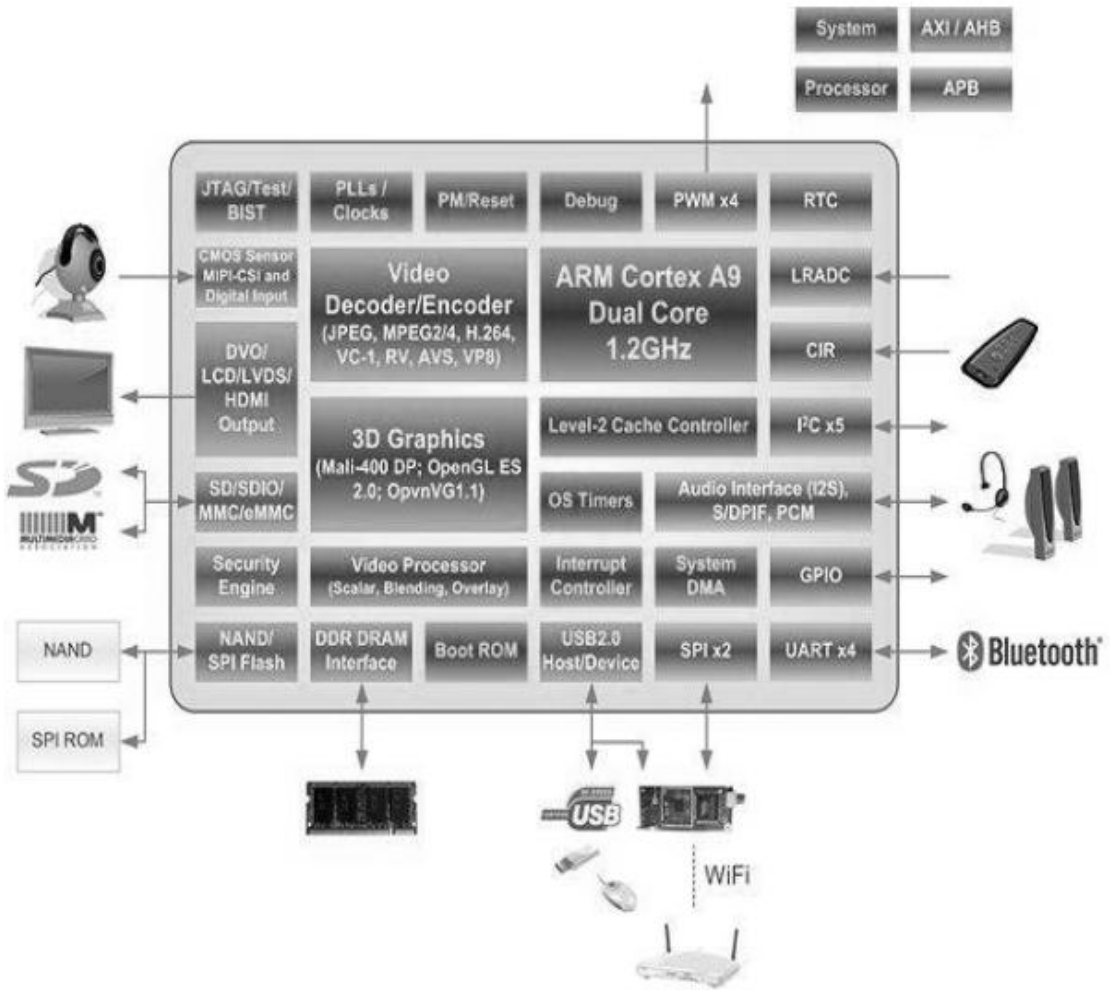
;

-

.

,

.



1.1 –

IP-

IP-

«

»

:

-

-

,

.

,

1.2

[1].

(RTL),

RTL-

Verilog VHDL. RTL-

RTL-

C

1.2.

:

RTL-

[3].

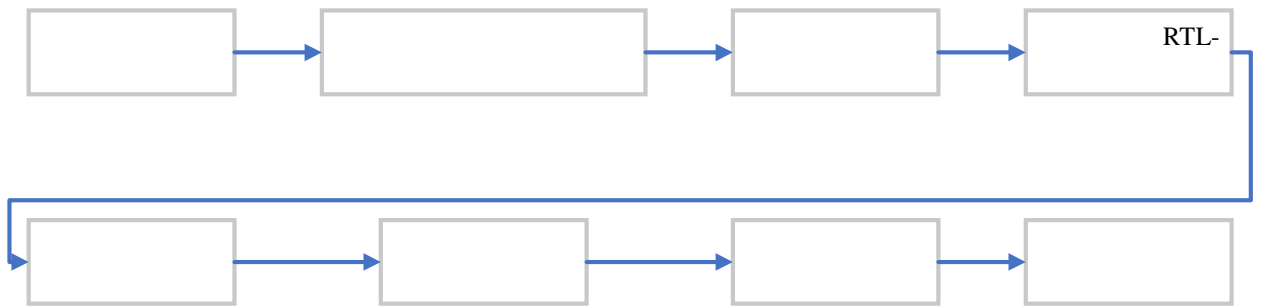
[4-7].

[3]:

- , ,
 , Verilog SystemVerilog;
 - (C / C ++, Java)
 ;
 -
 .
 , ;
 -

SystemC SystemVerilog.

[8],



1.2 –

1.3

, ,
 -

[11].

RTL-

RTL

RTL

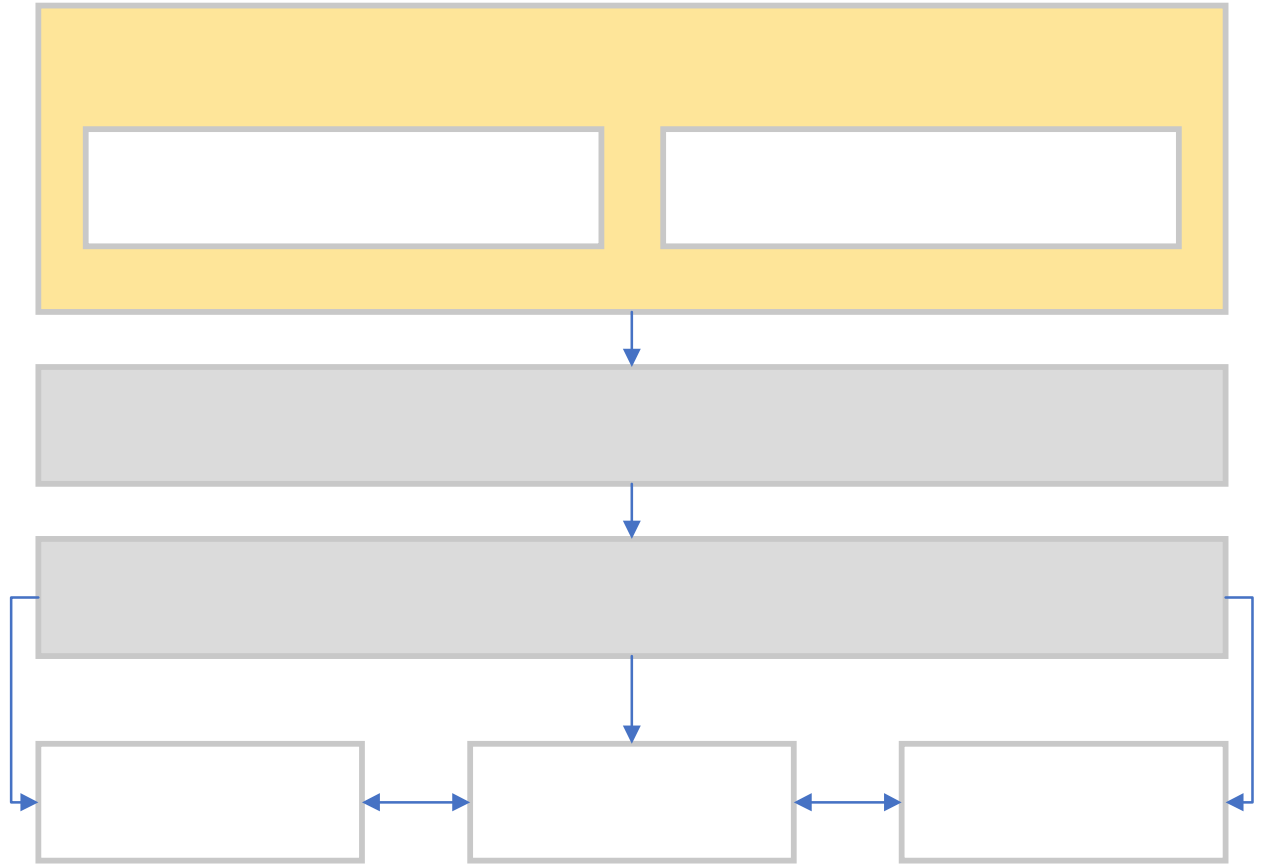
RTL-

RTL-

RTL

RTL-

1.3.



1.3 –

RTL-

,
RTL-

[1].

1.4

2

(). ,
 , :
 (VHDL, Verilog, SystemC) -
 (, - Altera HDL . .).
 , « - -
 » ASIC-
 . , ,
 ,
 . - ,
 RTL- Verilog VHDL- ,
 ().
 HDL-

2.1

[1]

« - », :
 - -

$\Sigma = \{T, X, U, \Omega, Y, \Gamma, \{, y\}$
 $(t_1, t_2]$
 $x(t_1)$
 $x(t_2)$
 $t_1 <$
 $t_2)$.

(1.1):

$$\Sigma = \{T, X, U, \Omega, Y, \Gamma, \{, y\} \tag{2.1}$$

$T \subseteq \mathbb{R};$
 $X = \{x : T \rightarrow \mathbb{R}^n\};$
 $U = \{u : T \rightarrow \mathbb{R}^m\};$
 $Y = \{y : T \rightarrow \mathbb{R}^p\};$
 $\Gamma : T \times T \times X \times U \rightarrow X;$
 $x(t) = (t; x_0, t_0), \quad x_0 \in X,$
 $t_0 \in T;$
 $\Omega : T \times X \rightarrow Y;$

$(t_0, t_1]$

$(t_0, t_1], t_0, t_1 \in T.$

$\langle t, x \rangle.$

$T \times X.$

[1], $T, X, Y,$

:

- T

;

-

$z : z :$

$$'(t) = (t +), \forall t, \in T;$$

-

$$s, (t; , x,) = (t + s; + s, x, zS,);$$

-

t.

(

$$) \quad T = R (T = Z).$$

2.2

-

-

,

,

,

,

,

,

,

,

.

-

,

,

,

.

,

,

,
 .
 ,
 ,
 ,
 .
 -
 ,
 ,
 -
 .

(2.2):

$$A = \{T, Z, X, \dots, Y, H, G\} \tag{2.2}$$

$$T = [0, T_f] \subset \mathbb{R} - \dots \tag{}$$

$$Z - \dots \tag{}$$

$$z = (z_1, z_2, \dots, z_l), \quad z_1, \dots, z_l - \dots \tag{}$$

$$\{Z(t)\} - \dots \tag{}$$

$$\{Z(0)\} - \dots \tag{}$$

$$X - \dots (x = [x^{(1)}, \dots, x^{(l)}]), \quad l - \dots \tag{}$$

$\{t_j\}, t_j \in T.$

$$\langle t_j, x_j \rangle, x_j \in X$$

$$- \dots (g = [g^{(1)}, \dots, g^{(r)}], \quad r - \dots \tag{}$$

$\{i\}, i \in T.$

$\langle i, g_i \rangle, g_i \in$

;

Y -

.

$\{k\}, k \in T,$

k

$z(t)$

$\{Z_y\},$

Z

Z.

$\langle k, y_k \rangle, y_i \in Y$

.

H -

,

;

G -

,

$$y(t) = G[z(0), t].$$

,

H

,

$z(0)$

$z(t)$

.

,

.

-

.

(\quad , \quad) .

...

-

.

,

,

.

.

, ... ,
 ... -
 .
 - ...
 (2.3):

$$S = \{T, P, e, E, K, F\}, \tag{2.3}$$

T = {ti}, ti ∈ R - ;
 P - ;
 e - ();
 E - ().

,
 K = {< ti, ei >, L} - ,
 .
 L ().

F - ,
 .
 (2.4):

$$P = \{X, Y, V_s, V_d, B\}, \tag{2.4}$$

X, Y - ;
 Vs - ,

$V_d - \dots$ « - » - ,
 F.
 -
 . . . ,
 ,
 .
 (),
 ,
 .
 .
 . . . , . . . , «
 » - ,
 .
 (,) ,
 ().
 (2.4):

$$H = \{S, X, E, F, \dots, \dots\}, \tag{2.4}$$

$S - \dots ;$
 $X - \dots ;$
 $E - \dots , \quad e = \langle s, a, e, (s_0),$
 $s_0 \in E, s, s_0 \in S - \dots \quad e, a \in \dots -$
 ();
 $- \dots X, \dots E;$
 $- \dots X \dots ;$
 $F - \dots , \dots ;$

– X , X
 $s \in S$ (s).

$x \in X$, $s \in S$,
 ($s(x)$). ,
 $\langle s, x \rangle$, $s \in S, x \in X$.

(1.6):

$$= \{T, Q, Q_0, Q_F, TR\}, \tag{2.6}$$

$T = \{t_i\}$ – ;
 Q – , Q_0, Q_F – X ;
 TR – ,
 : () ()
 . .) ()
).

[3],

(TESS, Model Vision for Windows, Statemate
 MAGNUM, i-Logic Rhapsody, HyTech),

-
 ;
 - ()
)

2.3

VHDL

VHDL,

(IEEE)

1987

[10],

1987 1993

real,

VHDL

IEEE 1076.6.

VHDL

().

:

;

```

-          (
-          );
-          (
-          ,
-          );
-          (
-          -
-          '
-          );
-          (for generate / if
generate);
-          (
-          );
-          assert / report.
-          .
-          ;
-          ;
-          ;
-          (
-          );
-          assert / report;
-          wait.

```

1987 1993 .

1999 VHDL

IEEE 1076.1 - VHDL AMS (Analog and Mixed

Systyems),

,
 ,
 .
 ()
 ,
 - VHDL-AMS
 across quantity, - through quantity.
 ' across through
 ' , -
 . ,
 ,
 -
 VHDL-AMS ,
 .
 - ,
 VHDL-AMS
 ,
 . - ,
 ,
 ,
 - ,
 ,
 .
 , , , , ,
 quantity, real. dX/dt
 X $X'Dot,$ X 0
 - $X'Integ.$
 , (simultaneous

statement), quantity. simultaneous
 statement : 'dot = * + b * .

, ,
 <=.

(energy domain)

(package).

nature

2.1.

```
Nature electrical is
voltage across
current through
electrical_ref reference;
```

2.1 – - VHDL-AMS

voltage – , a current – .

across, through

reference temperature, heatflow, termal_ref . . .

entity

1.2.

```
entity unit is
port (terminal x1, x2: electrical;
terminal g1, g2, g3: fluidic);
end unit;
```

2.2 – - VHDL-AMS

```

x1, x2 - , g1, g2, g3 -
.
/ ,
, quantity.
: port (quantity input: in real; quantity output: out real);
quantity ,
: quantity v across i 1, i2 through node_a to node_d;
, v -
node_a node_d, , il i2
, .
break.
TReference
TContribution. TReference across quantity
(ground). TContribution - through quantity ,
. TReference
.
, simultaneous statement
. quantity
(real) .
, VHDL-
- .

```

3

(HDL), VHDL

Verilog,

Internal Intermediate Representation (IIR) [7],
 AIRE / CE (Advanced Intermediate Representation
 with Extensibility / Common Environment).

HDL - EDA, :

(VHDL

Verilog),

IIR;

HDL -

IIR;

IIR;

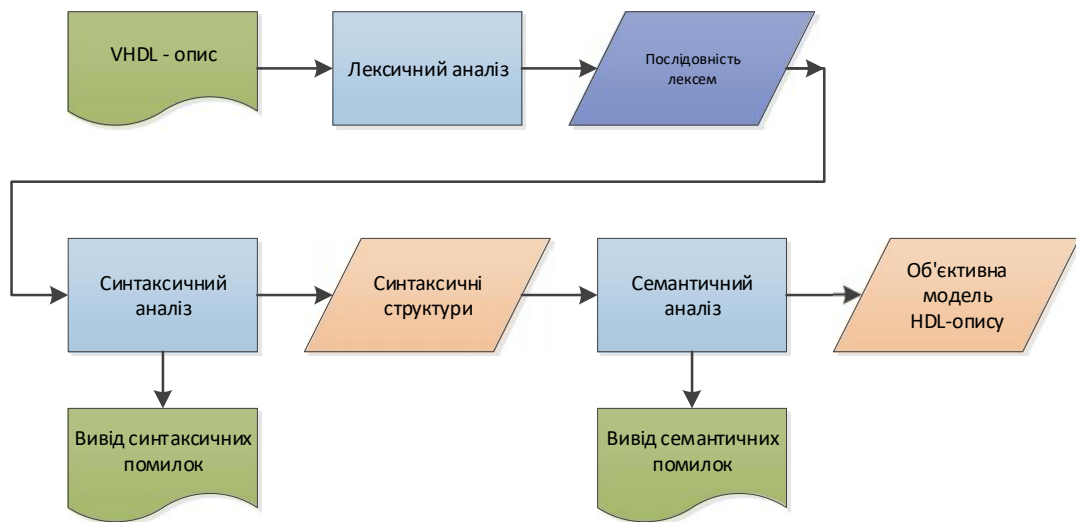
2- :

IIR /

VHDL Verilog

HDL -

(3.1).



3.1 – VHDL-

3.1

- (. Top-down parser) -
- (LL-);
- (. Bottom-up parser) -

(LR-

,
, GLR-).

(LL)

- ,

, . LL- LL (k)

- , k

()

LL (k) - , LL

(k) - .

LR- -

(Right)

LR (k) - , k

, . k

1

GLR (. Generalized Left-to-right Rightmost derivation

parser -) -

LR- ,

. GLR

LR

, , GLR

. GLR

LR

LR

(
) , GLR

GLR

/

/

ANTLR (. Another Tool For Language Recognition - «
 ») - ,
 - (C ++,
 Java, C #, Python, Ruby) LL (*) - ,
 - [12].
 Coco / R, ANTLR, LL (1) ,
 , LL
 (1) , - ,
 ,
 ANTLR
 C #. ANTLR
 2- :
 2
 3.2
 « »
 (token). ,
 , .
 .
 : < - ,
 - >. ANTLR,
 (
), .
 , ,

2.1.

```

BASIC_IDENTIFIER
  :   LETTER ( LETTER_OR_DIGIT | '_' )* ;

//extended identifiers can't contain a single backslash
EXTENDED_IDENTIFIER :   '\\\ ' ( '\\\ ' | '\\\ \ ' | '%\ ' | GRAPHIC_CHARACTER
)+ '\\\ ' ;

```

2.1 –

VHDL

ANTLR

: Stream). (-

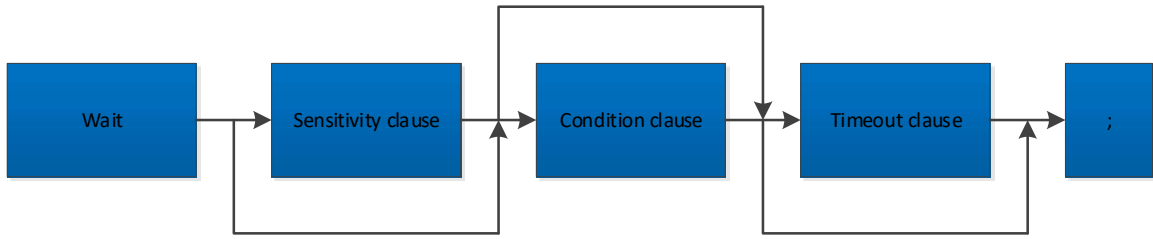
3.3

()

(). -

wait VHDL

(3.2).



3.2 –

wait

ANTLR

3.2.

```

wait_statement:
  WAIT          sensitivity_clause?   condition_clause?
  timeout_clause? SEMI
  ->          ^ (WAIT          sensitivity_clause?   Condition_clause?
  Timeout_clause?);
  
```

3.2 –

wait

VHDL

,

,

,

..

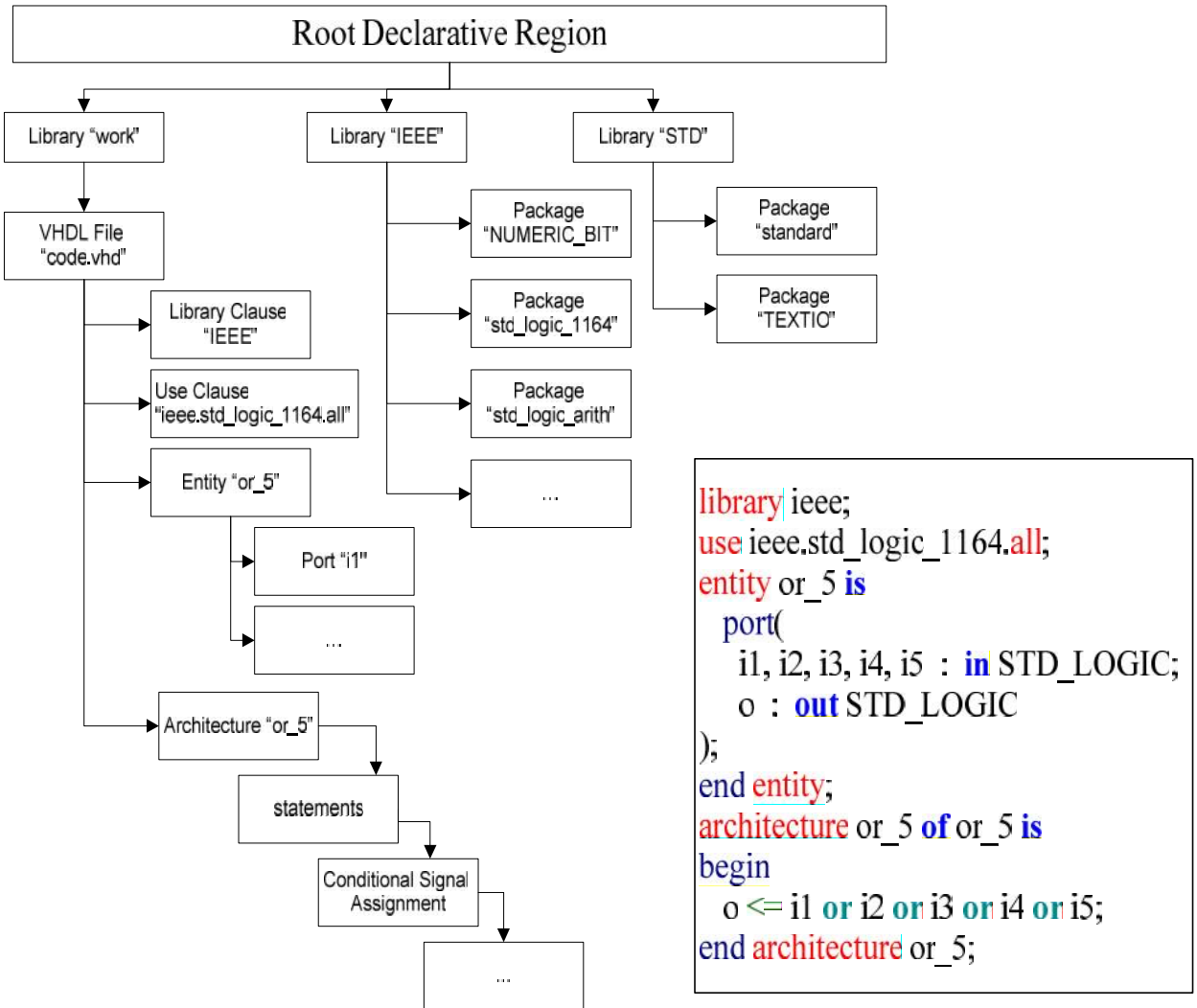
(2.3).

3.4

,

,

.



3.4 –

options (output, ASTLabelType, language, superClass).

2.4
VHDL-

3.5

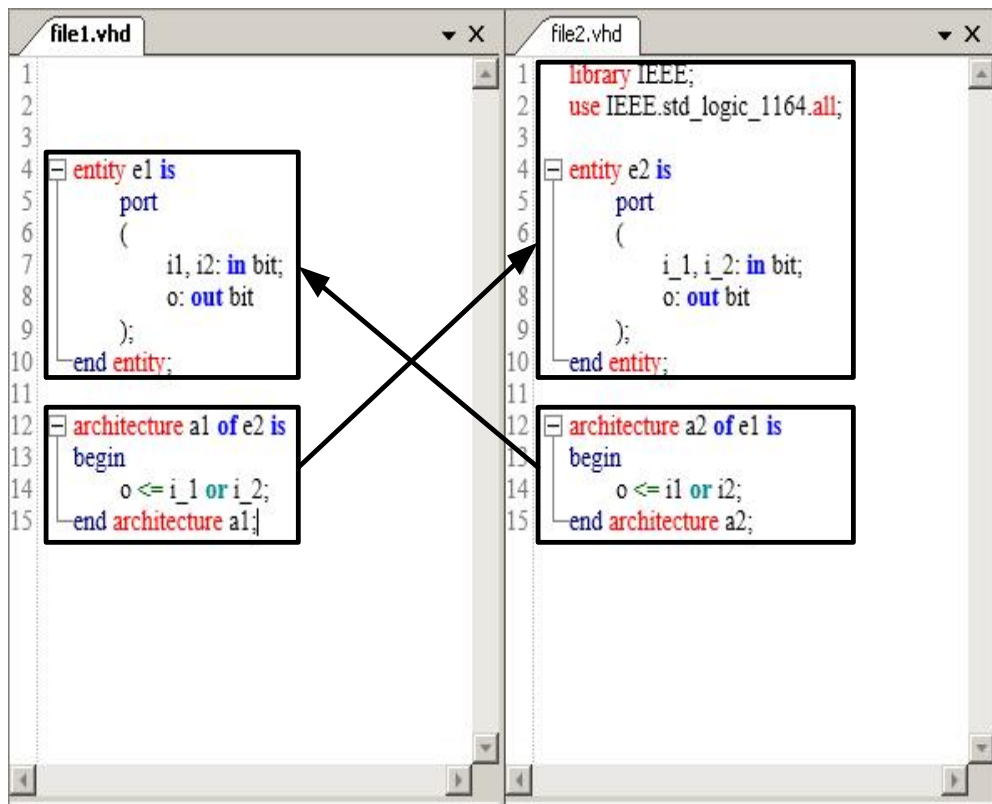
(primary unit) (secondary unit),
VHDL

HDL- (entity,
)
()
)

```

:
-
, use,
;
- secondary unit,
primary unit, ' ( ' - entity,
- ).

```



3.5 –

architecture-entity

VHDL

File1. File2 File4 (File2 file3.
). File5, ' - File6.
 , ' «
 entity e2».

3.6

VHDL -

IntelliSense -

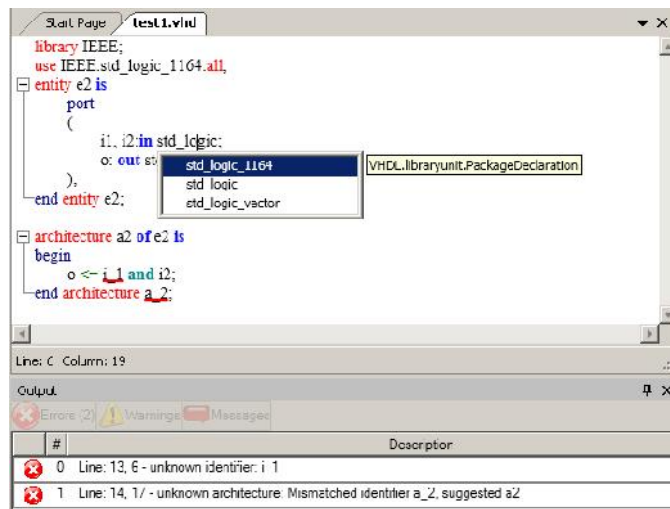
Microsoft

- Microsoft Visual Studio.

VHDL Verilog)

<< >>, << >>

(3.7).



3.7 –

4

,
 .
 ,
 ,
 ,
 ,
 .

4.1

,
 .
 ,
 .
 ,
 ,
 ,
 ,
 ,
 .

[19].

,
 .
 :
)
 .
 ,
 ,
 ,

(Unified Modelling Language) [17],

UML.

4.1

UML-

[20].

()

2.3

UML-

N M [21].

UML-

- BPMN (Process Model

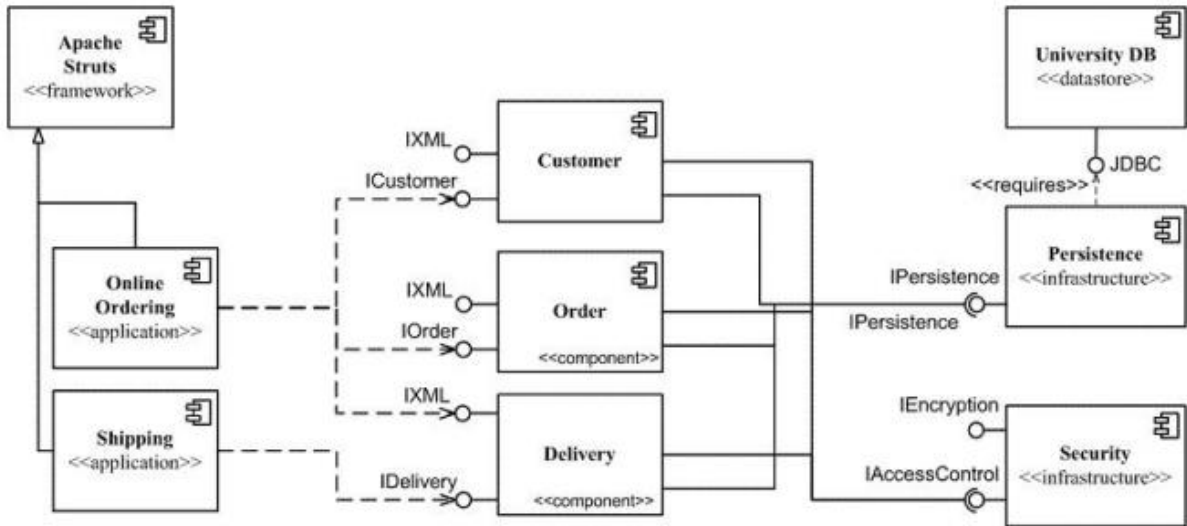
Notation) [22].

BPMN

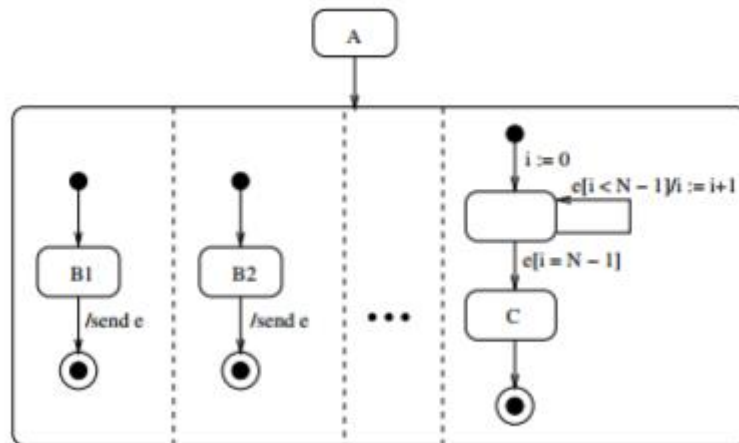
UML

BPMN

3.3)



4.1 – UML-



4.2 – UML-

UML,

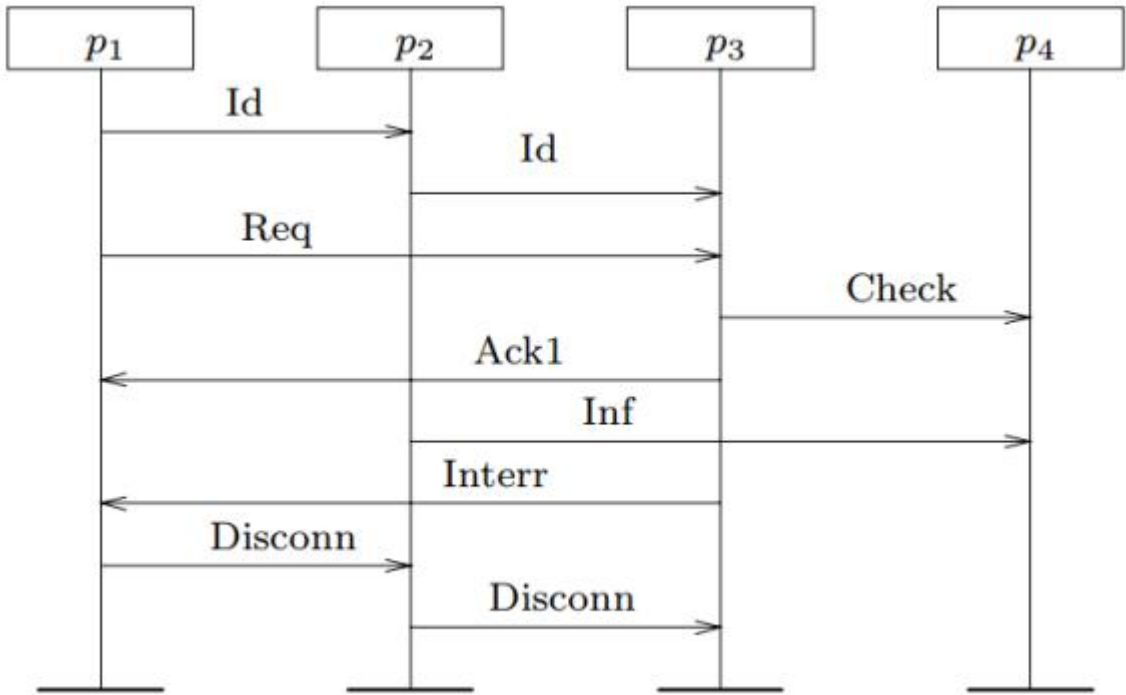
(MSC,

Message Sequence Charts).

2.3

[23].

UML



4.3 –

) . ,

.

.

,

,

,

:

- ;

- ;

- ;

- (UML- BPMN);

(UML- , ,

,

);

- .

4.2

,

· ,

·

·

,

,

·

,

,

,

·

·

,

,

·

·

·

,

·

4.3

,

:-

,

; -

; -

.

,

(

,

,

,

),

,

: -

,

,

; -

,

,

.

,

.

,

,

,

.

,

.

,

,

.

,

,

,

.

.

,

.

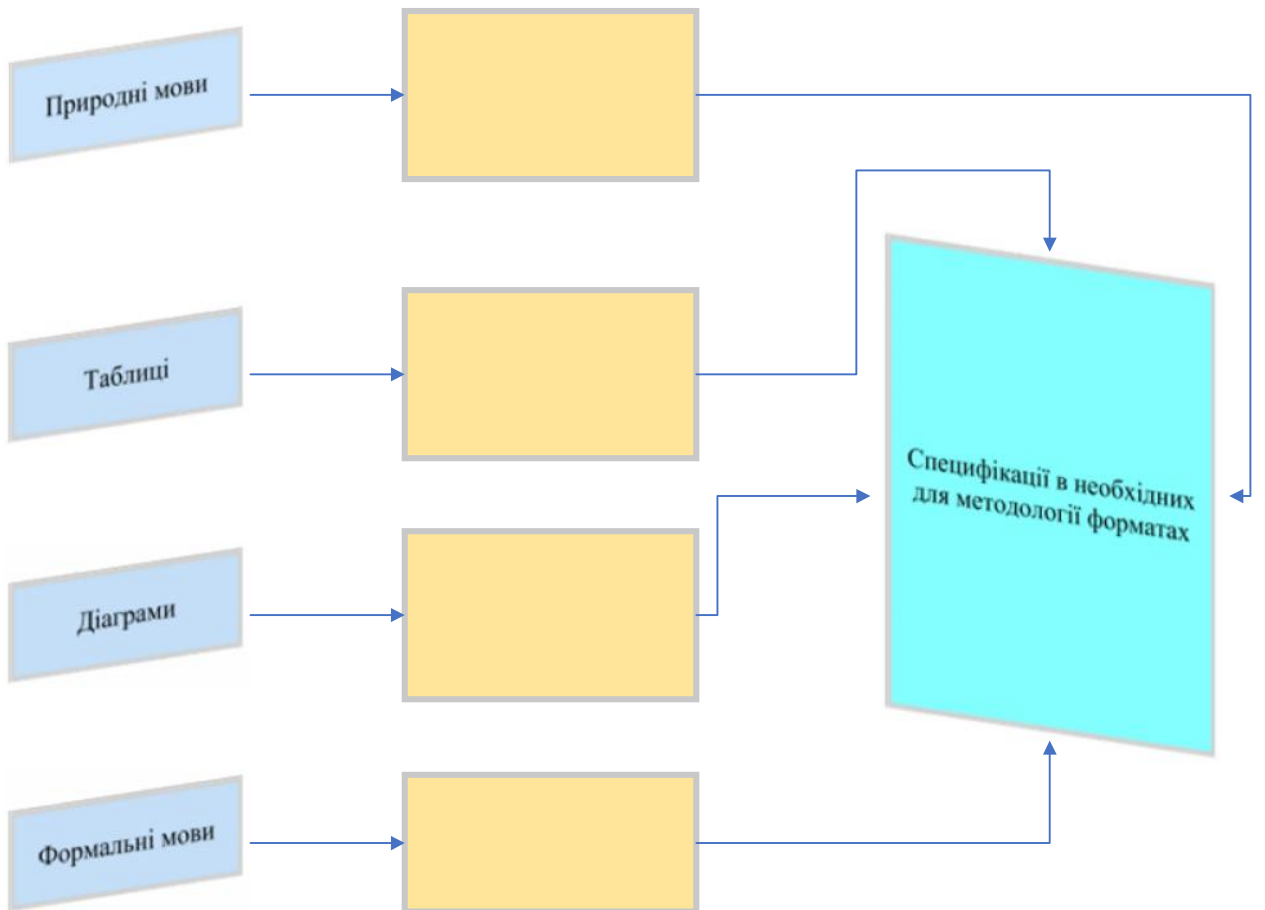
,

.

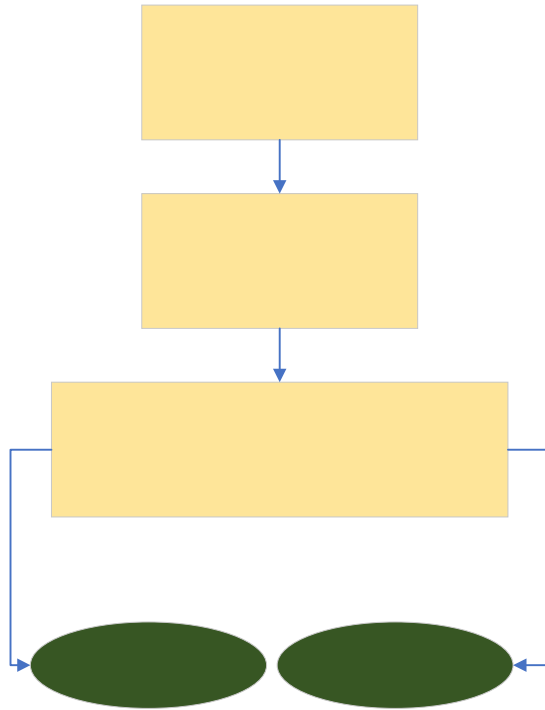
,

,

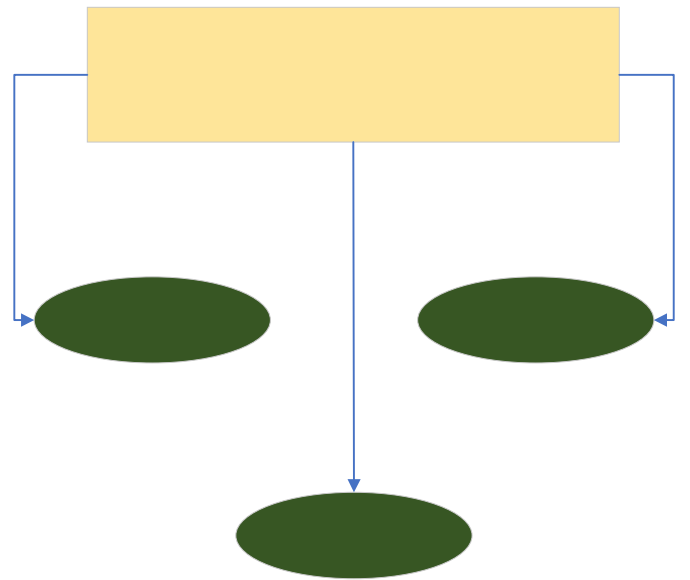
-



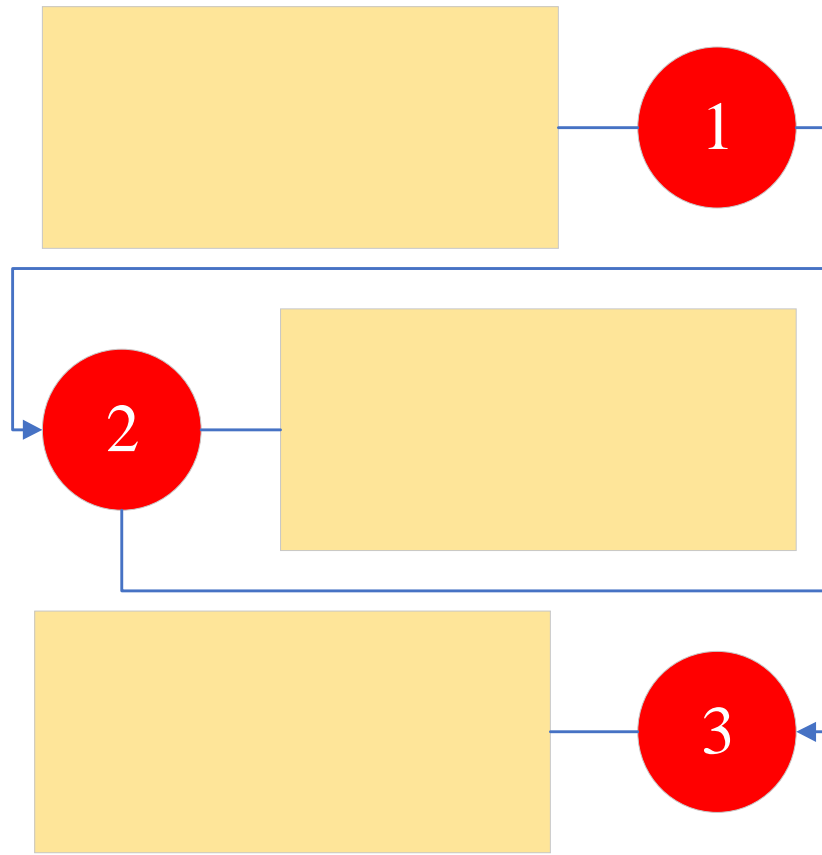
4.5.



4.5 –



(4.6)



4.6 –

:

-

-

-

-

4.4

,

,

,

,

,

.

Microsoft Office.

Microsoft Excel,

,

-

,

Microsoft Visio.

,

,

,

,

,

,

,

IFlow

3.

.

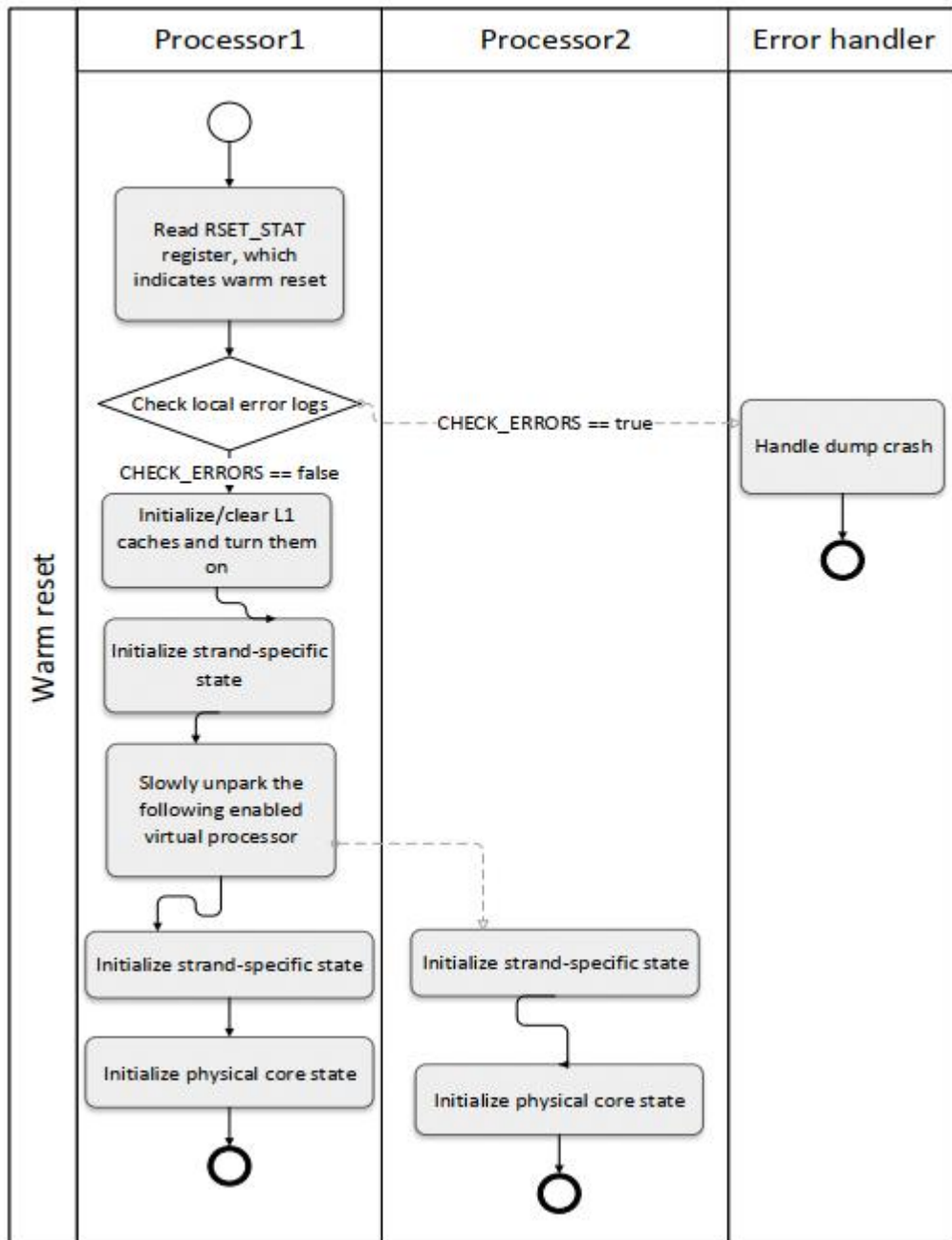
,

IFlow

```

<assert>
<expr>v1!=v2</expr>
<var><name>v1</name><table_type>RegisterTable</table_type>
<column>Name</column>
<row>[1:-1]</row> </var> <var> <name>v2</name> <column>Name</column>
<row>[1:-1]</row></var>
</assert>

```



4.7 –

```

        "RegType"
        ,
        "FieldName"
        ,
        "RegType".
        :
        "BitWidth",
        "Access",
    
```

"BitOffset",

-

"ResetVal"

RDL-

Register Field Definitions							
Comments							Reset
FieldName	RegType	Bit Offset	Bit Width	Description	Access	ResetVal	
-	PSTATE	13	51	Reserved	RO	0	
tct	PSTATE	12	1	Trap on control transfer	RW	0	
-	PSTATE	10	2	Reserved	RO	0	
cle	PSTATE	9	1	Current little endian	RW	0	
tle	PSTATE	8	1	Trap little endian	RW	0	
-	PSTATE	6	2	Reserved	RO	0	
-	PSTATE	5	1	Reserved (was red)	RO	0	
pef	PSTATE	4	1	Enable floating-point	RW	1	
am	PSTATE	3	1	Address mask	RW	0	
priv	PSTATE	2	1	Privileged mode	RW	1	
ie	PSTATE	1	1	Interrupt enable	RW	0	
-	PSTATE	0	1	Reserved (was ag)	RO	0	

4.8 –

Register Field Definitions							
Comments							Reset
FieldName	RegType	BitOffset	BitWidth	Description	Access	ResetType	ResetVal
gl	TSTATE	40	2	Global level at previous trap level	RW	PowerUp	0
ccr	TSTATE	32	8	CCR at previous trap level	RW	PowerUp	0
asi	TSTATE	24	8	ASI at previous trap level	RW	PowerUp	0
pstate tct	TSTATE	20	1	PSTATE.tct at previous trap level	RW	PowerUp	0
pstate cle	TSTATE	17	1	PSTATE.cle at previous trap level	RW	PowerUp	0
pstate tle	TSTATE	16	1	PSTATE.tle at previous trap level	RW	PowerUp	0
pstate pef	TSTATE	12	1	PSTATE.pef at previous trap level	RW	PowerUp	0
pstate am	TSTATE	11	1	PSTATE.am at previous trap level	RW	PowerUp	0
pstate priv	TSTATE	10	1	PSTATE.priv at previous trap level	RW	PowerUp	0
pstate ie	TSTATE	9	1	PSTATE.ie at previous trap level	RW	PowerUp	0
cwp	TSTATE	0	3	CWP from previous trap level	RW	PowerUp	0
tct	PSTATE	12	1	Trap on control transfer	RW	PowerUp	0
cle	PSTATE	9	1	Current little endian	RW	PowerUp	0
tle	PSTATE	8	1	Trap little endian	RW	PowerUp	0
pef	PSTATE	4	1	Enable floating-point	RW	PowerUp	1
am	PSTATE	3	1	Address mask	RW	PowerUp	0
priv	PSTATE	2	1	Privileged mode	RW	PowerUp	1
ie	PSTATE	1	1	Interrupt enable	RW	PowerUp	0

4.9 –

```
reg {
name = "Processor State";
regwidth = 64;
reg {
name = "Processor State";
regwidth = 64;
reg {
name = "Processor State";
regwidth = 64;
field {
hw = rw; sw = rw;
fieldwidth = 1;
resetsignal = PowerUp;
reset = 0;
desc = Address mask;
} am [4:3]
```

4.2 –

:

,

,

.

.

;

.

1. [1] M. G. H. van den Broek, J. H. M. van den Broek, and J. H. M. van den Broek, "Requirements elicitation for systemC // Proceedings of the conference on Design, Automation and Test in Europe-Volume 2. – IEEE Computer Society, 2003. – . 6. – . 7-11.
2. WonderMedia PRIZM WM8950. [] URL: <http://www.wondermedia.com.tw/en/products/platform/soc/wm8950>
3. Riccobene E. et al. A SoC design methodology involving a UML 2.0 profile for SystemC // Proceedings of the conference on Design, Automation and Test in Europe-Volume 2. – IEEE Computer Society, 2005. – P. 704-709.
4. Lee Y. et al. Customer Requirements Elicitation based on Social Network Service // KSII Transactions on Internet & Information Systems. – 2011. – . 5. – . 10 .
5. Keller T. Contextual requirements elicitation // Seminar in Requirements Engineering, Spring 2011, Department of Informatics.
6. Dhungana D., Seyff N., Graf F. Research preview: Supporting end-user requirements elicitation using product line variability models // International Working Conference on Requirements Engineering: Foundation for Software Quality. – Springer Berlin Heidelberg, 2011. – . 66-71.
7. Kumari S. N., Pillai A. S. A survey on global requirements elicitation issues and proposed research framework // Software Engineering and Service Science (ICSESS), 2013 4th IEEE International Conference on. – IEEE, 2013. – P. 554-557.
8. P. [] / P. , , . - : , 1971. - 294 .
9. . : [] / . . . - : , 1978. - 488 .

10. [] / - : ", 1975. - 188 .
11. [] / . . . ; - : , 1989. - 314 .
12. . . . - [] / . . // . - 1997, 1. - . 36-52.
13. . . . [] / . . . - : , 1988. - 230 .
14. Internal Intermediate Representation (IIR) Specification Version 4.6 Including Digital VHDL & VHDL-AMS support [] / IEEE computer society, 2000. - 364 .
15. IEEE 1076-93 Standard VHDL Language Reference Manual [] / IEEE Computer Society, 1993. - 146 pp.
16. IEEE 1076-2008 Standard VHDL Language Reference Manual [] / IEEE Computer Society, 2008. - 254 pp.
17. Perry D.L. VHDL: Programming by Example [] / Douglas L.Perry. - McGraw-Hill, 2004. - 476 pp.
18. Black D.C. SystemC from the Ground Up [] / D.C.Black, J.Donovan. - Kluwer academic publishers, 2007. - 244pp.
19. Parr T. The definitive ANTLR reference [] / T.Parr. - The Pragmatic Programmers, 2009. - 396 pp
20. . windows communication foundation .NET Framework 3.5 [] / - Addison-Wesley, 2008. - 480 c.